

FIG 1

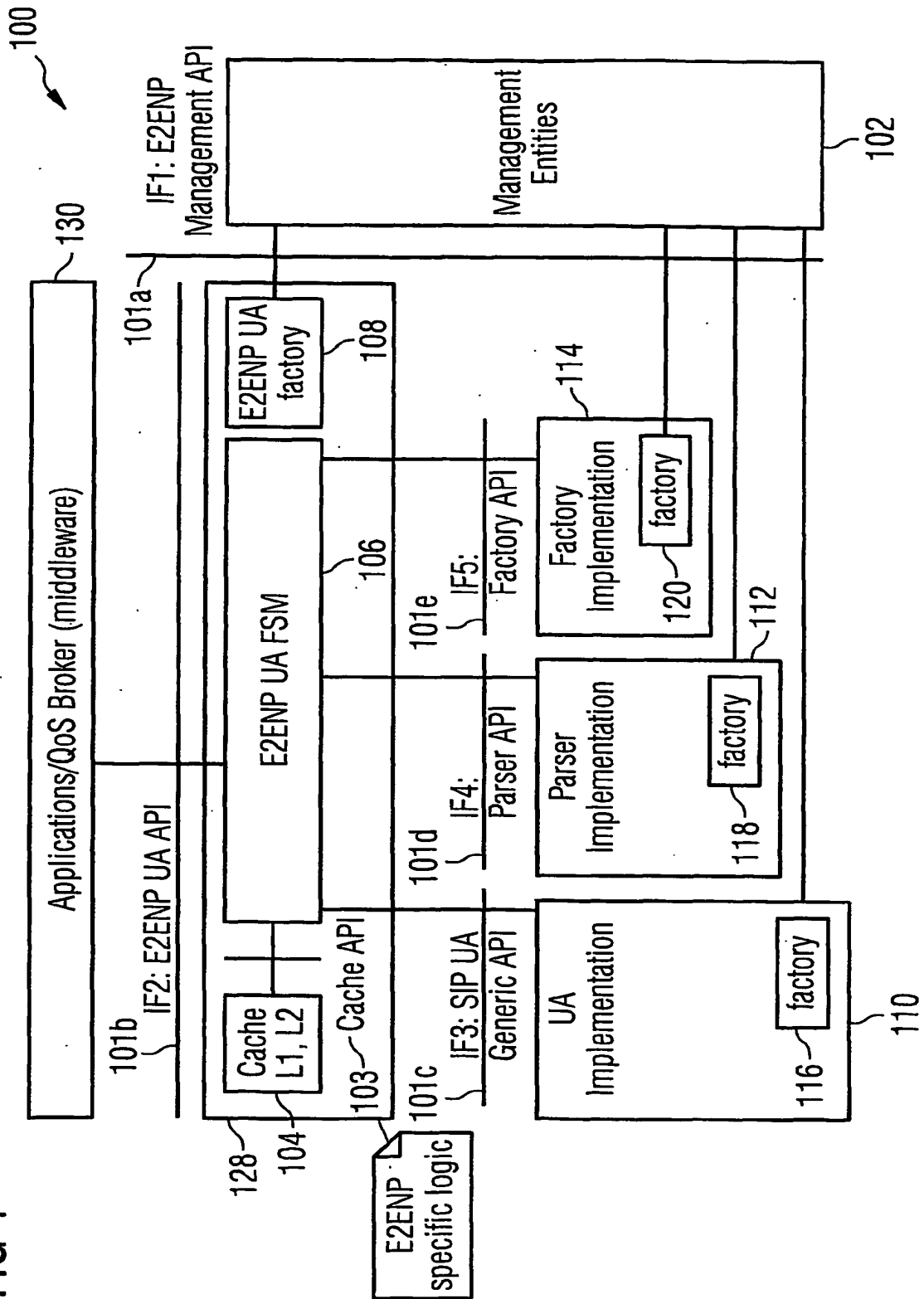


FIG 2

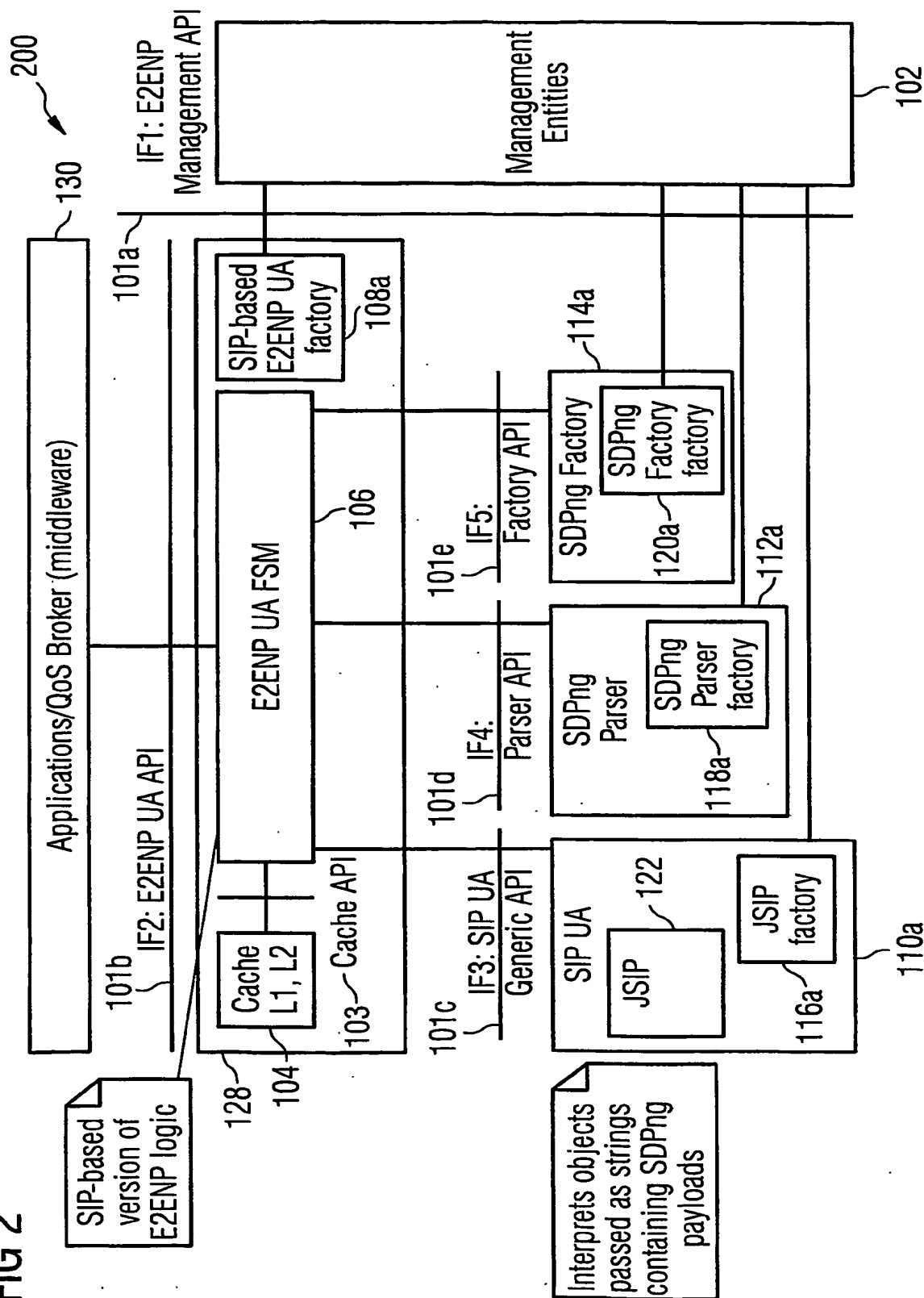


FIG 3

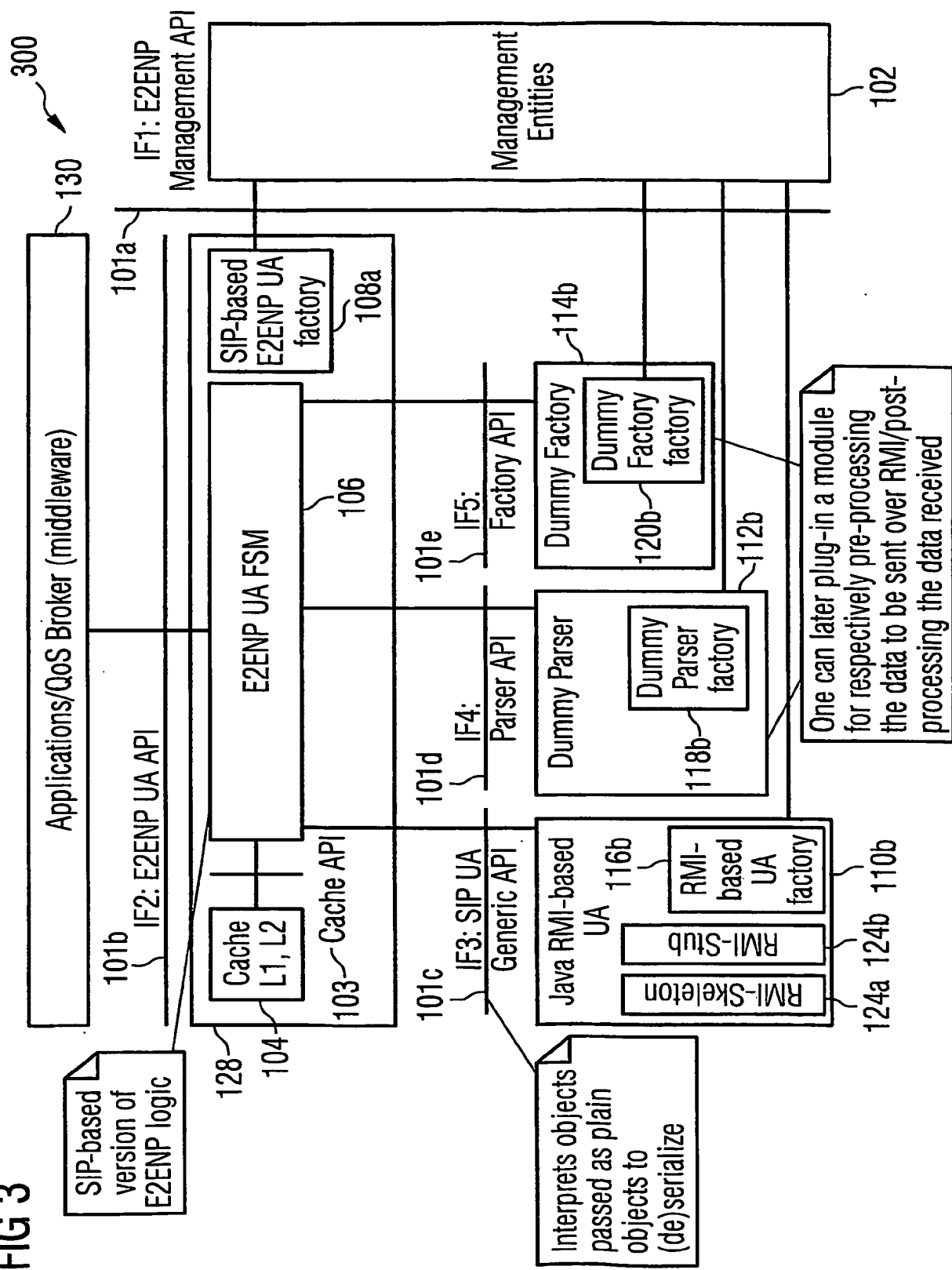
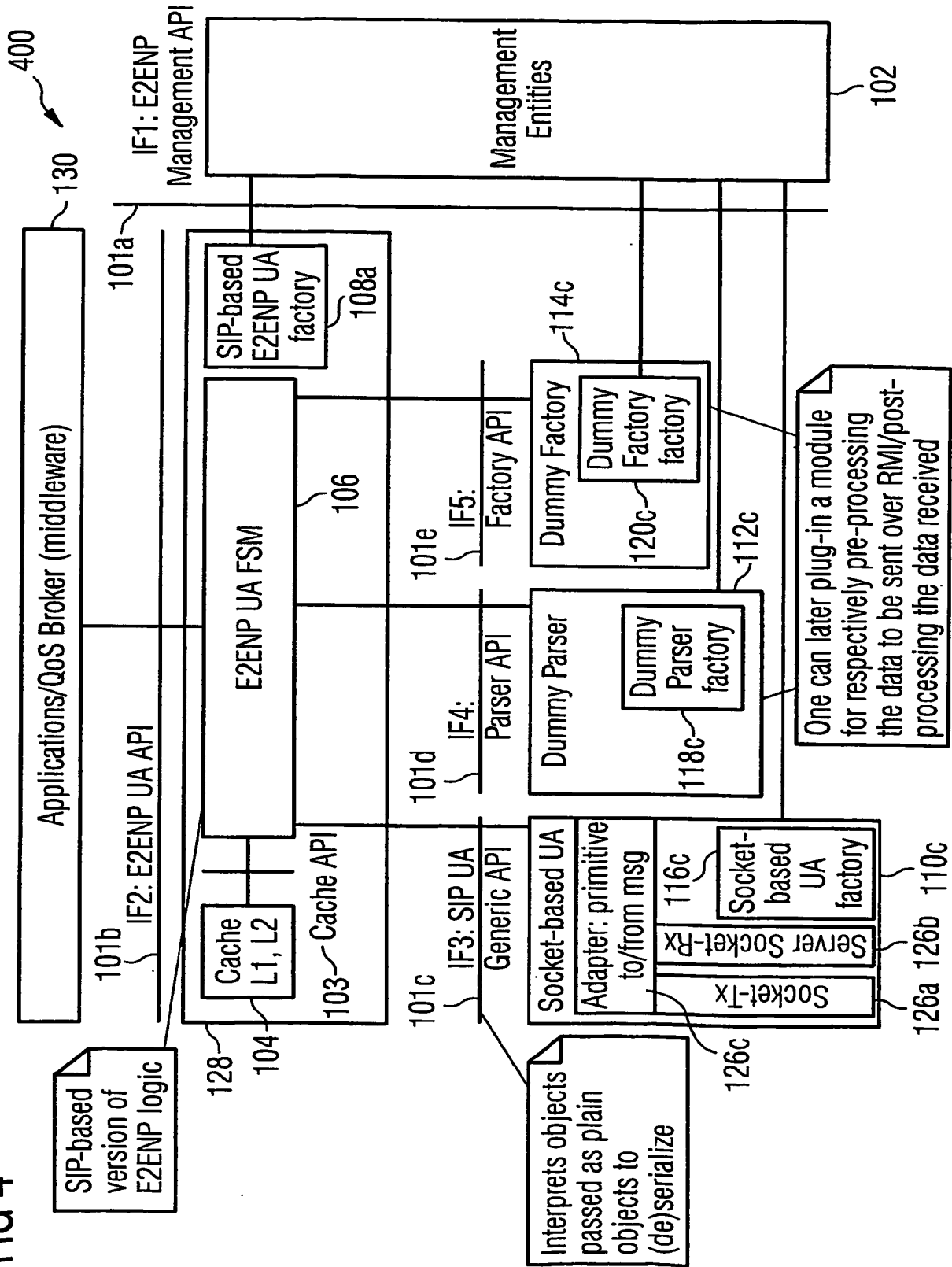


FIG 4



5/58

FIG 5

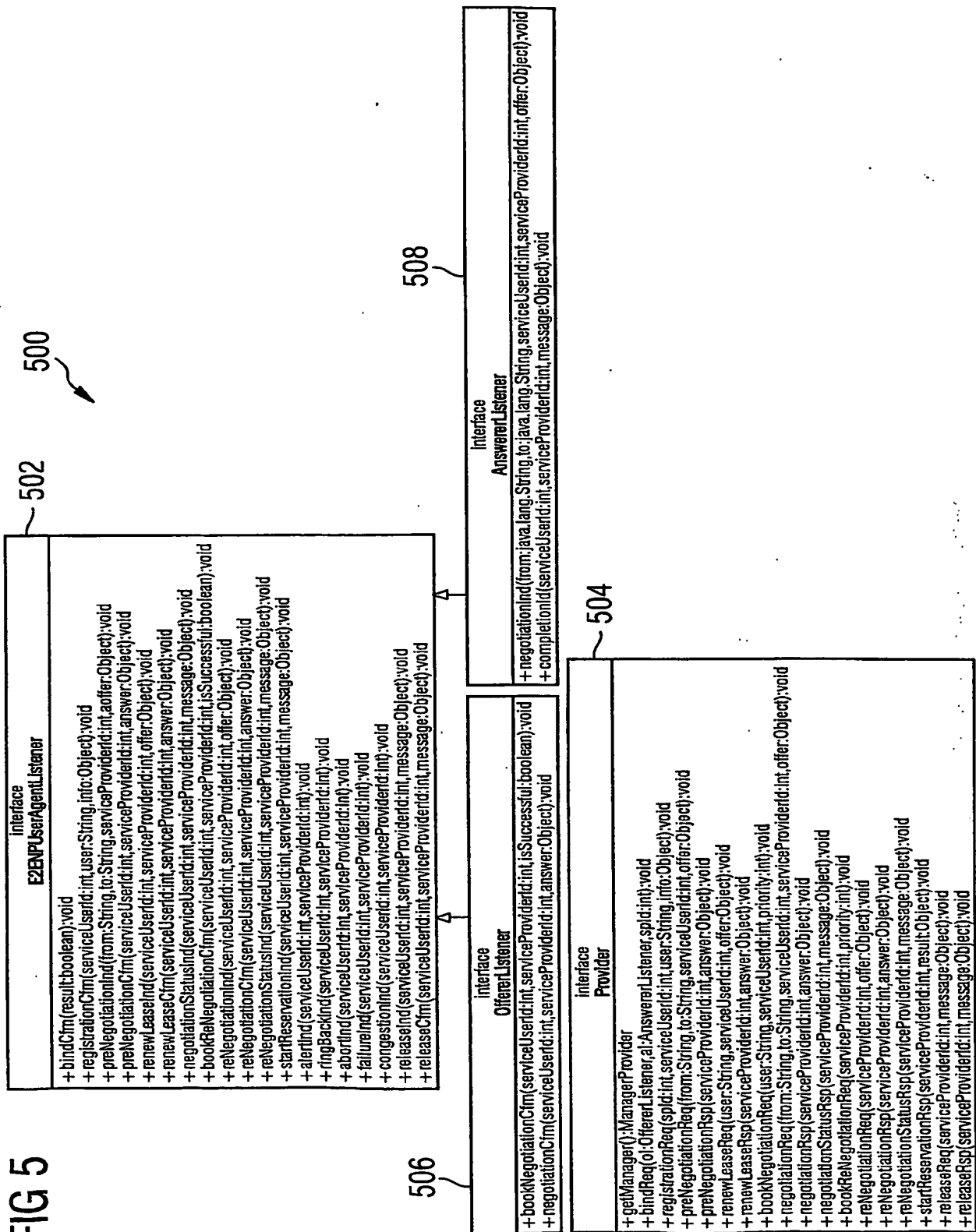
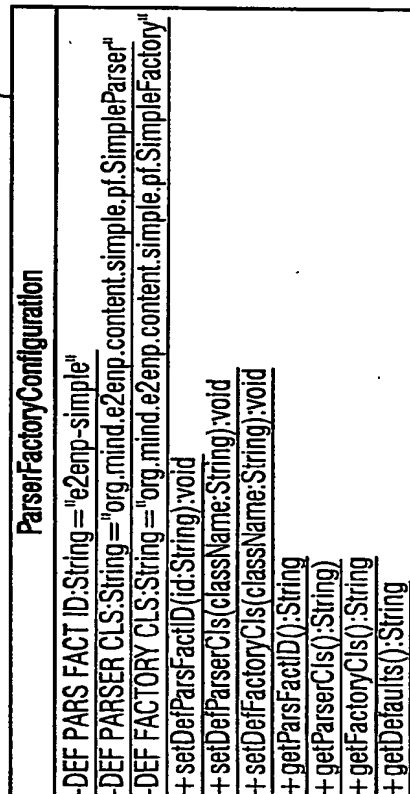
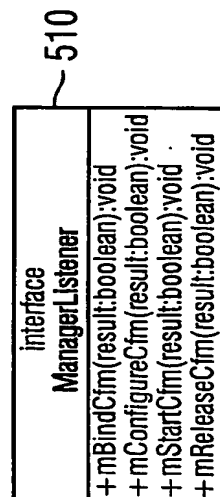
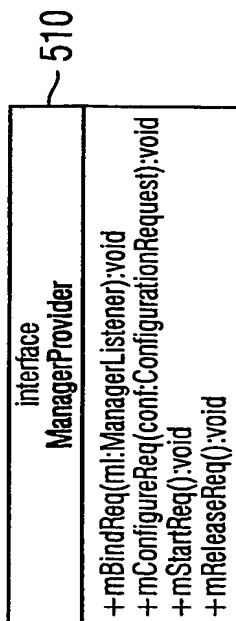


FIG 5bis



500

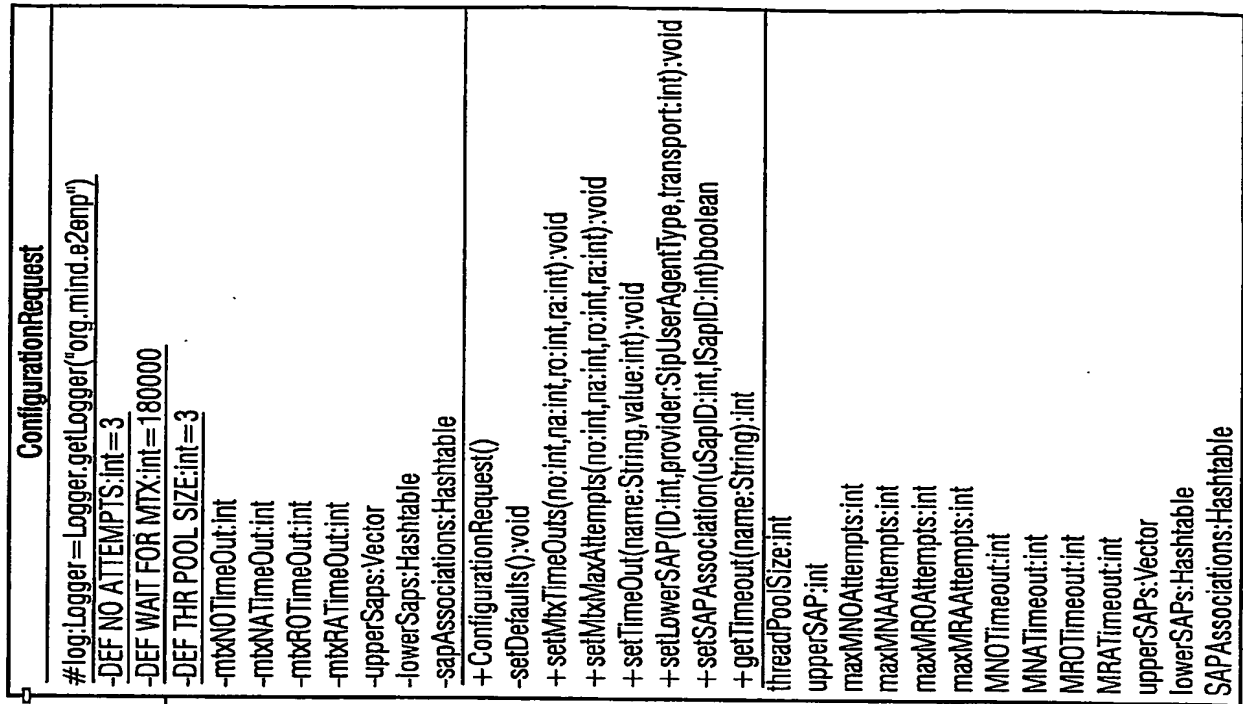
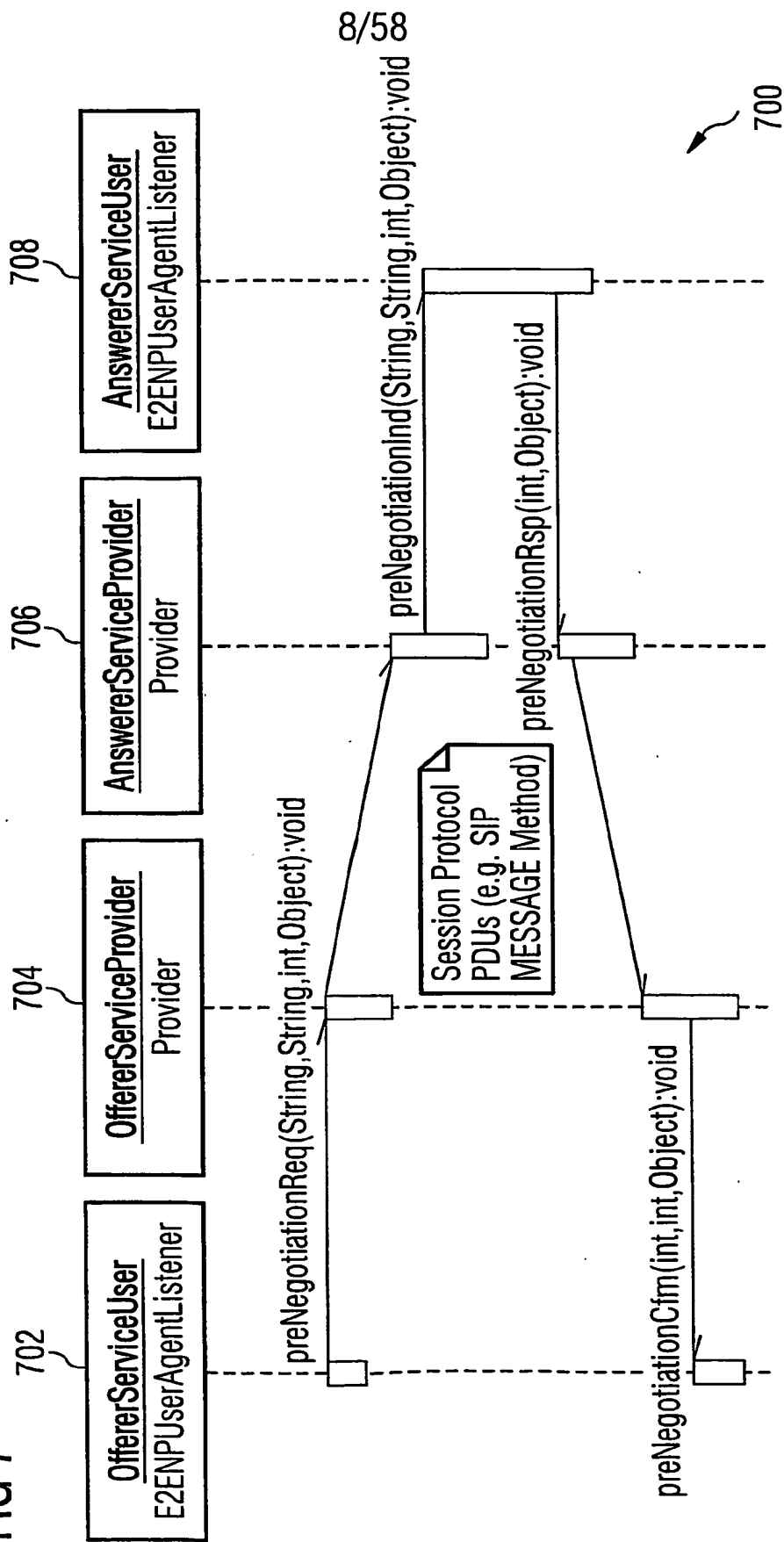
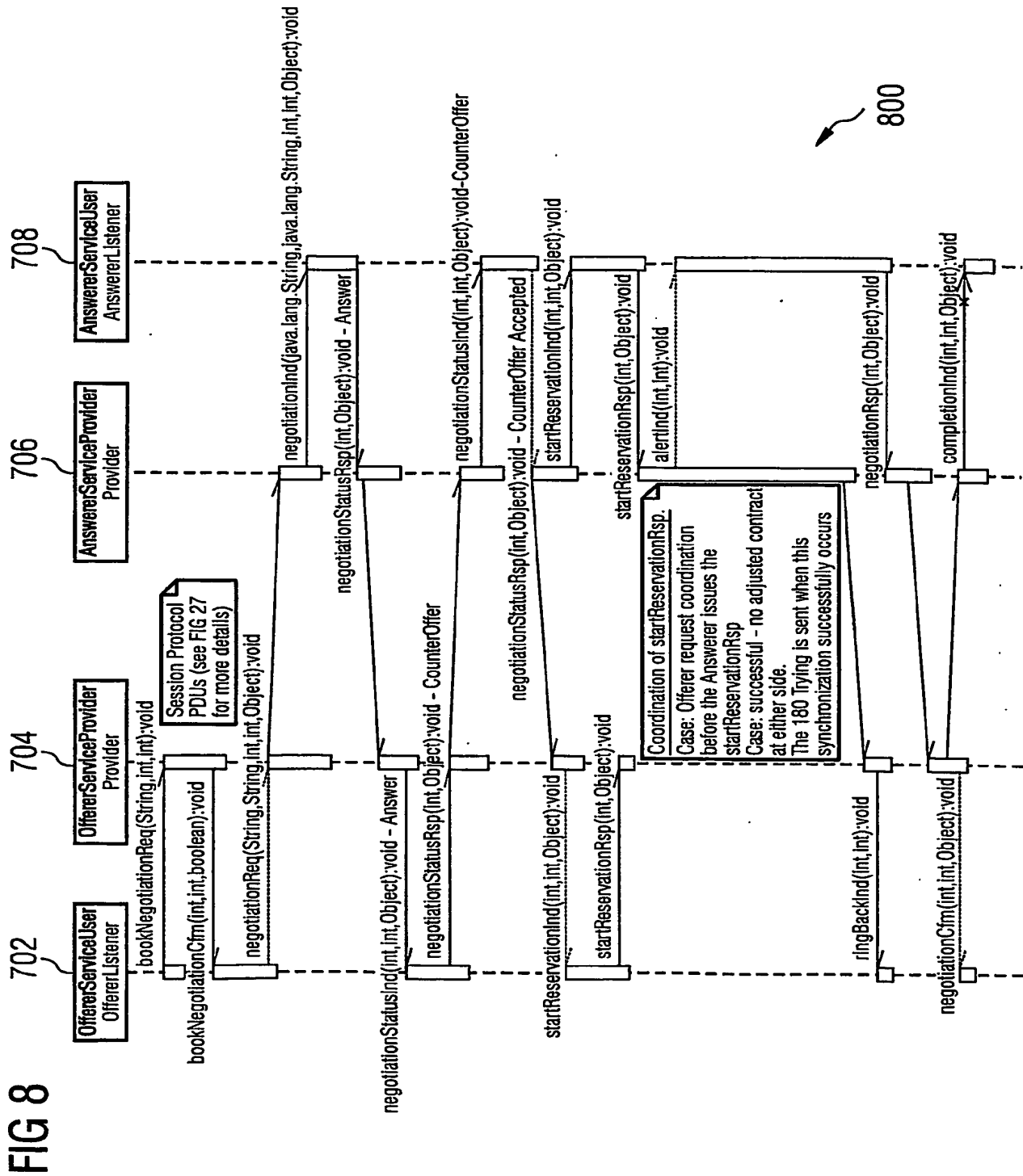


FIG 6

E2ENP-URI	=	"e2enp://" [userinfo "@"] hostport
userinfo	=	[user] [":" password]
user-unreserved	=	"&" / "=" / "+" / "\$" / "," / ";" / "?"
escaped	=	"%" HEXDIG HEXDIG
unreserved	=	alphanum / mark
mark	=	"_" / " " / "." / "!" / "~" / "*" / " " / "(" / ")"
alphanum	=	ALPHA / DIGIT
user	=	*(unreserved / escaped / user-unreserved)
password	=	*(unreserved / escaped / "&" / "=" / "+" / "\$" / ",")
hostport	=	host / host ":" port
host	=	hostname / hostaddress
hostname	=	*(domainlabel ".") toplabel ["."]
domainlabel	=	alphanum / alphanum * (alphanum / "-") alphanum
toplabel	=	ALPHA / ALPHA * (alphanum / "-") alphanum
hostaddress	=	ipv4-addr / ipv6-addr
ipv4-addr	=	digits "." digits "." digits "." digits
digits	=	1*3DIGIT
ipv6-addr	=	hexpart [":" IPv4address]
hexpart	=	hexseq / hexseq "::" [hexseq] / ":" [hexseq]
hexseq	=	hex4 *(":" hex4)
hex4	=	1*4HEXDIG
port	=	1*DIGIT

FIG 7





10/58

FIG 9

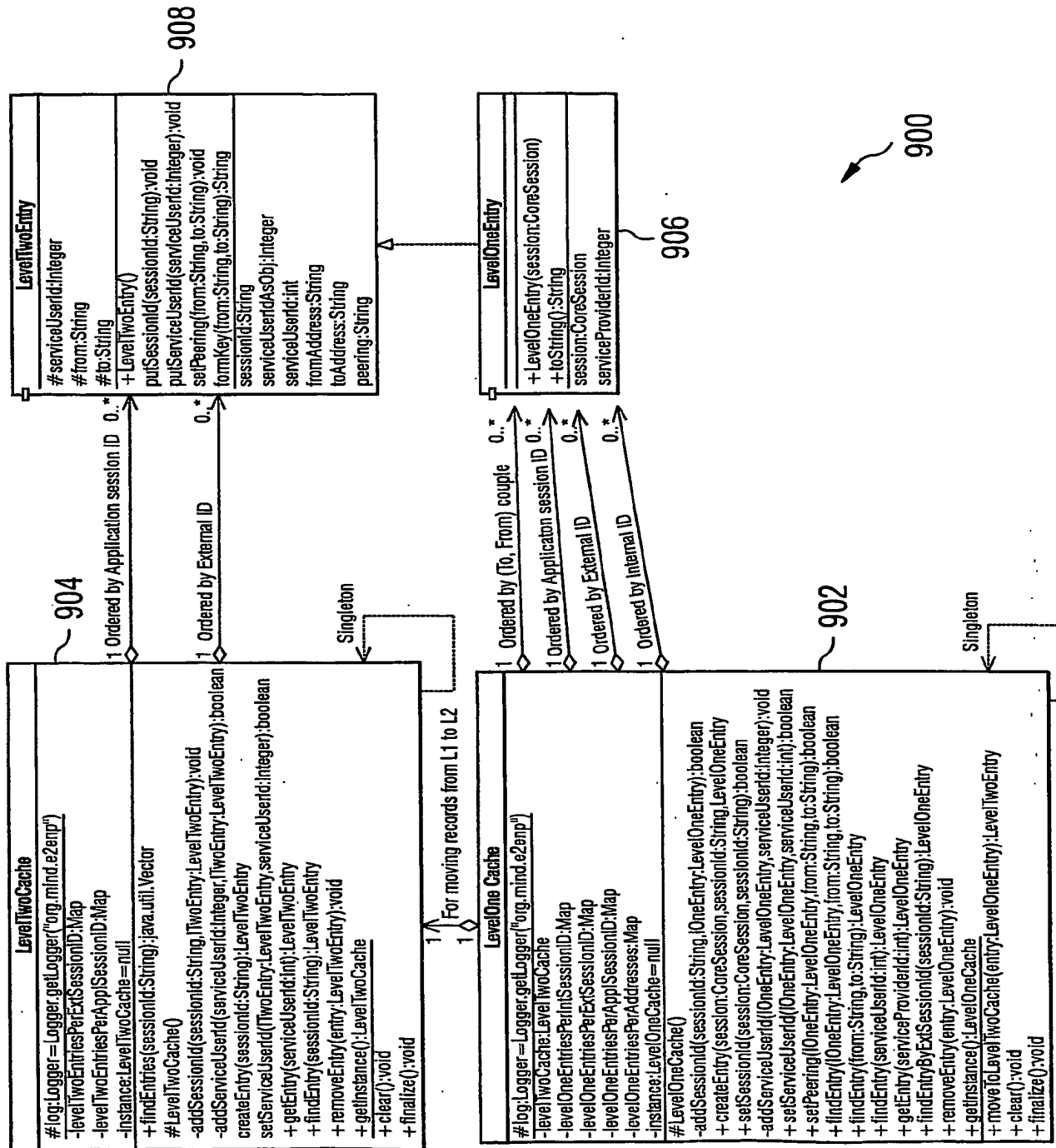
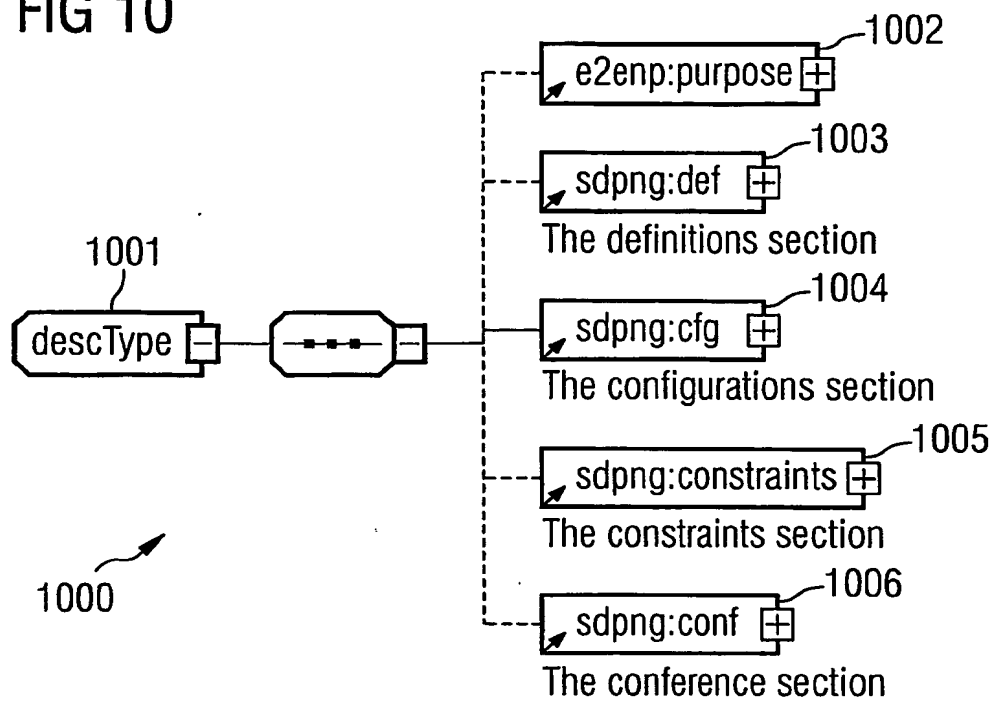


FIG 10



12/58

FIG 11

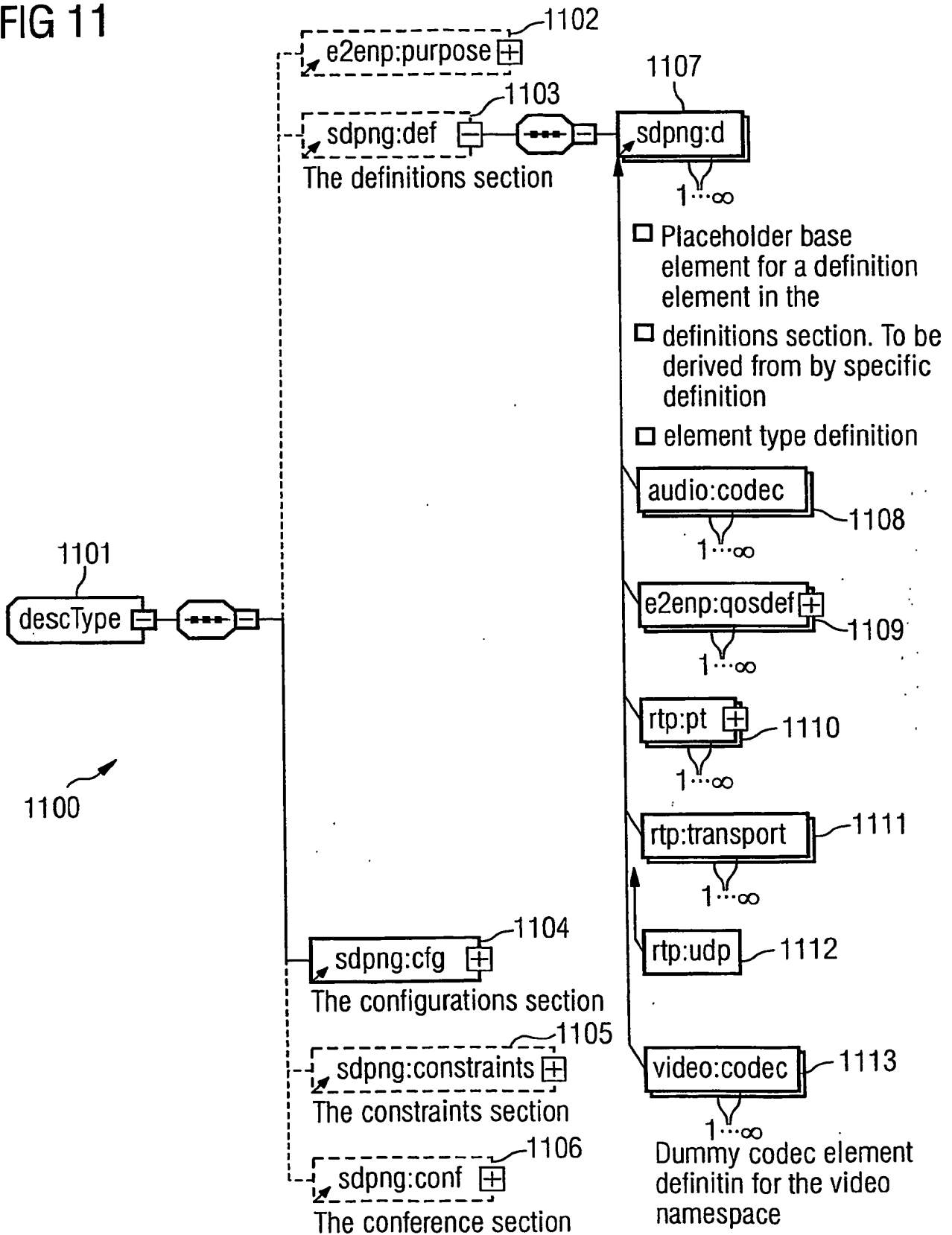
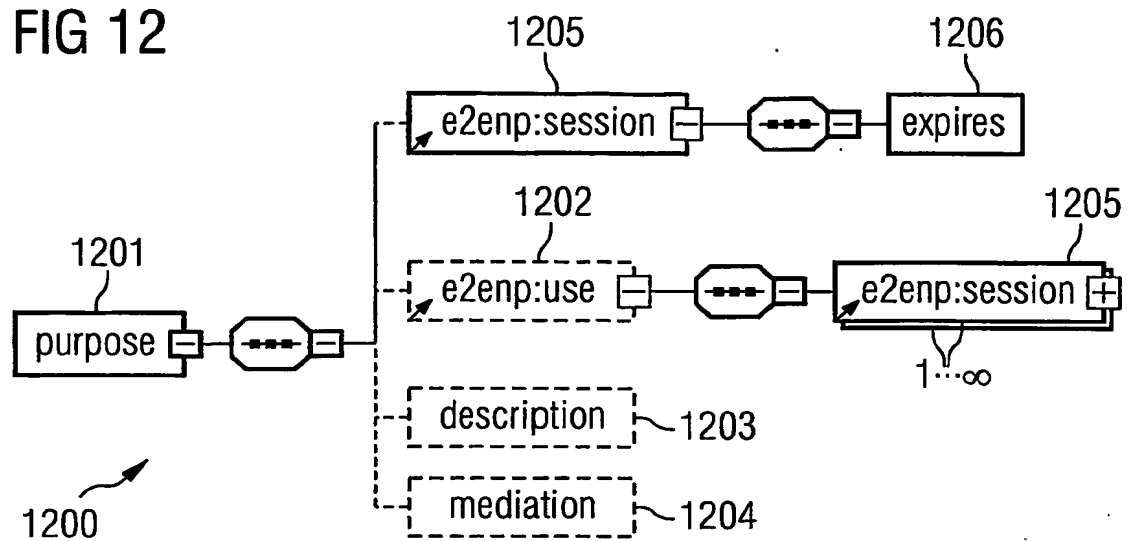


FIG 12



14/58

FIG 13

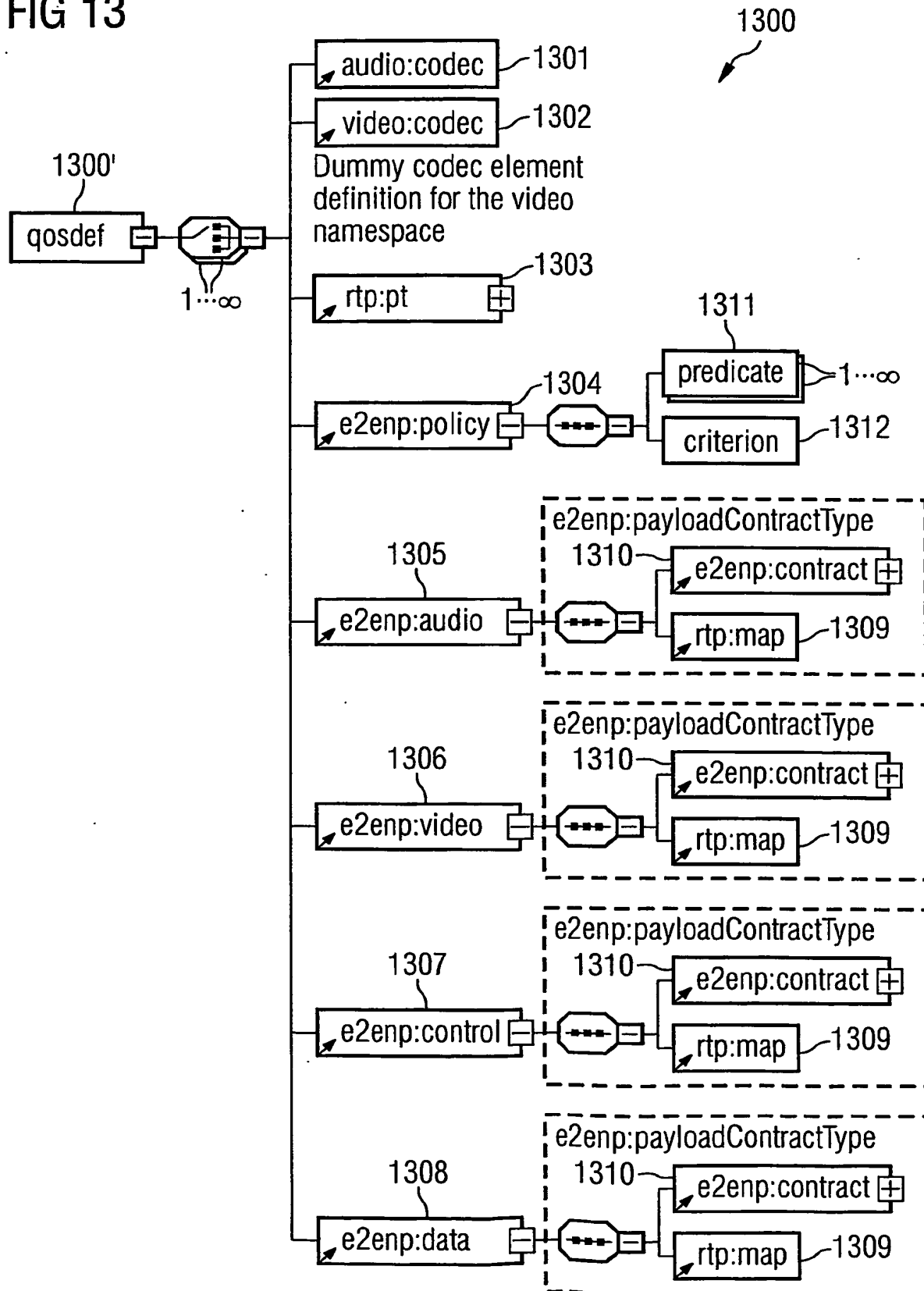
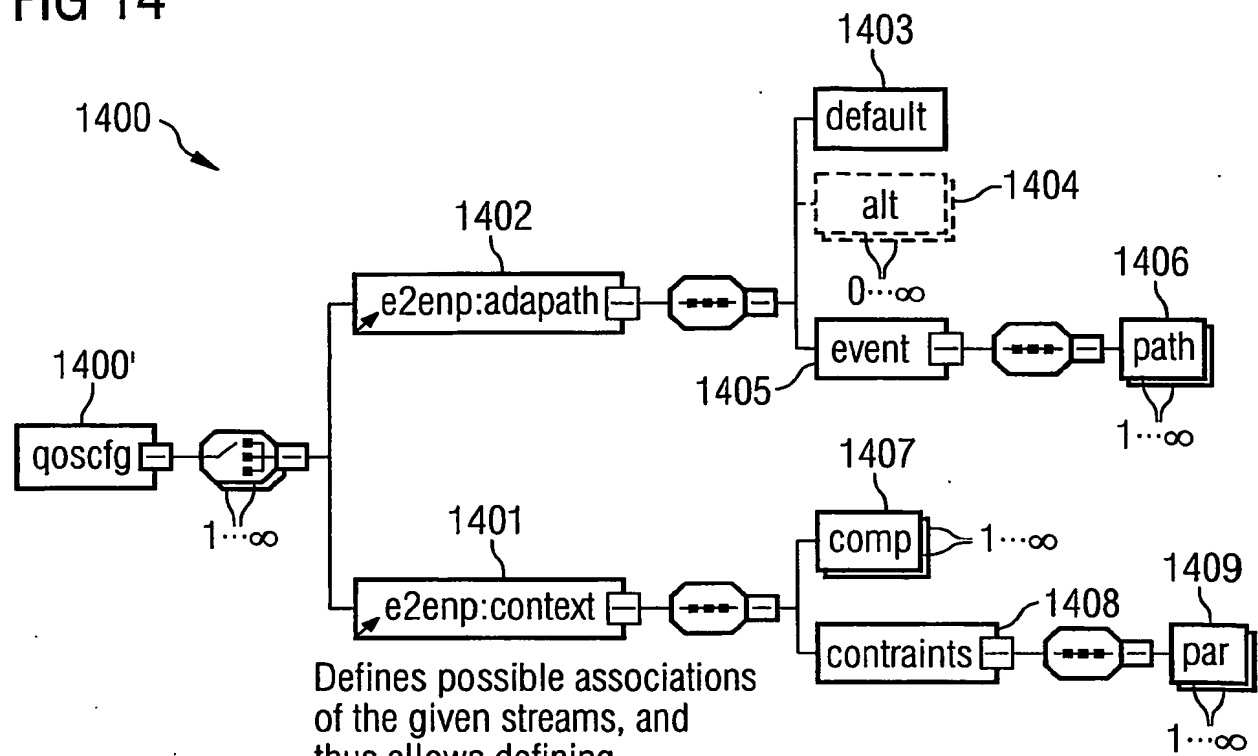
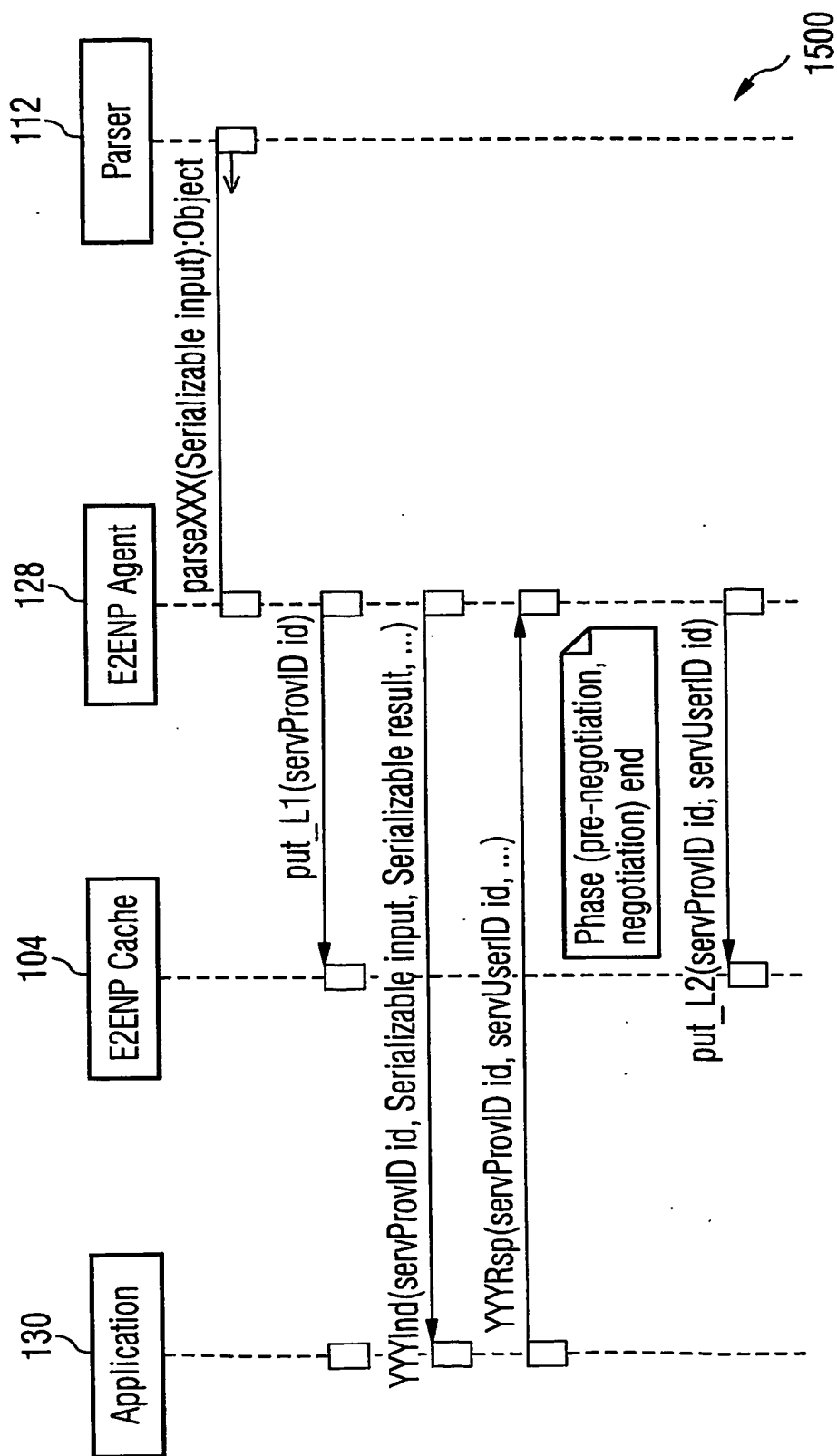


FIG 14



Defines possible associations of the given streams, and thus allows defining time-synchronization and/or QoS correlation constraints. As such, this element basically describes high-level QoS Contracts.

FIG 15



17/58

FIG 17

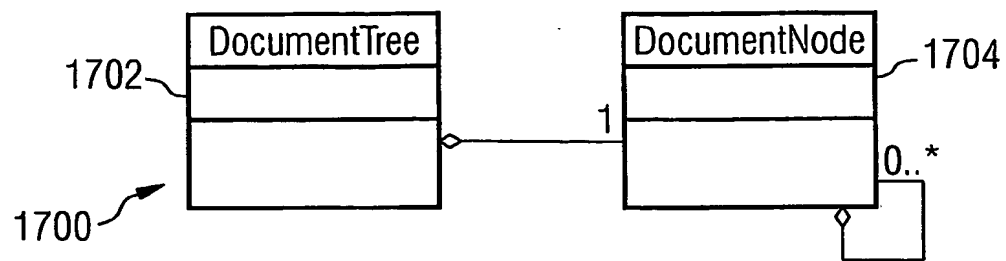
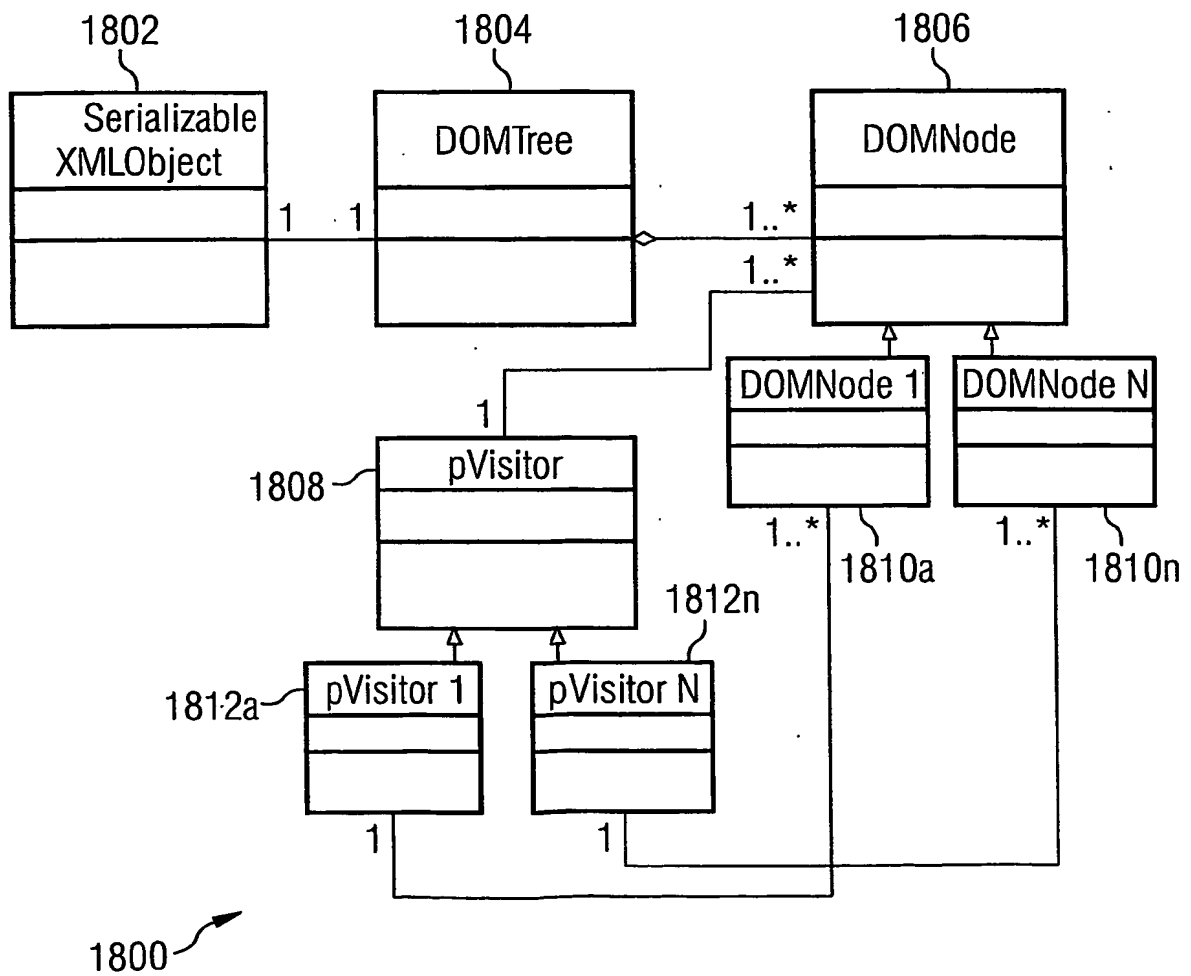


FIG 18



18/58

FIG 19

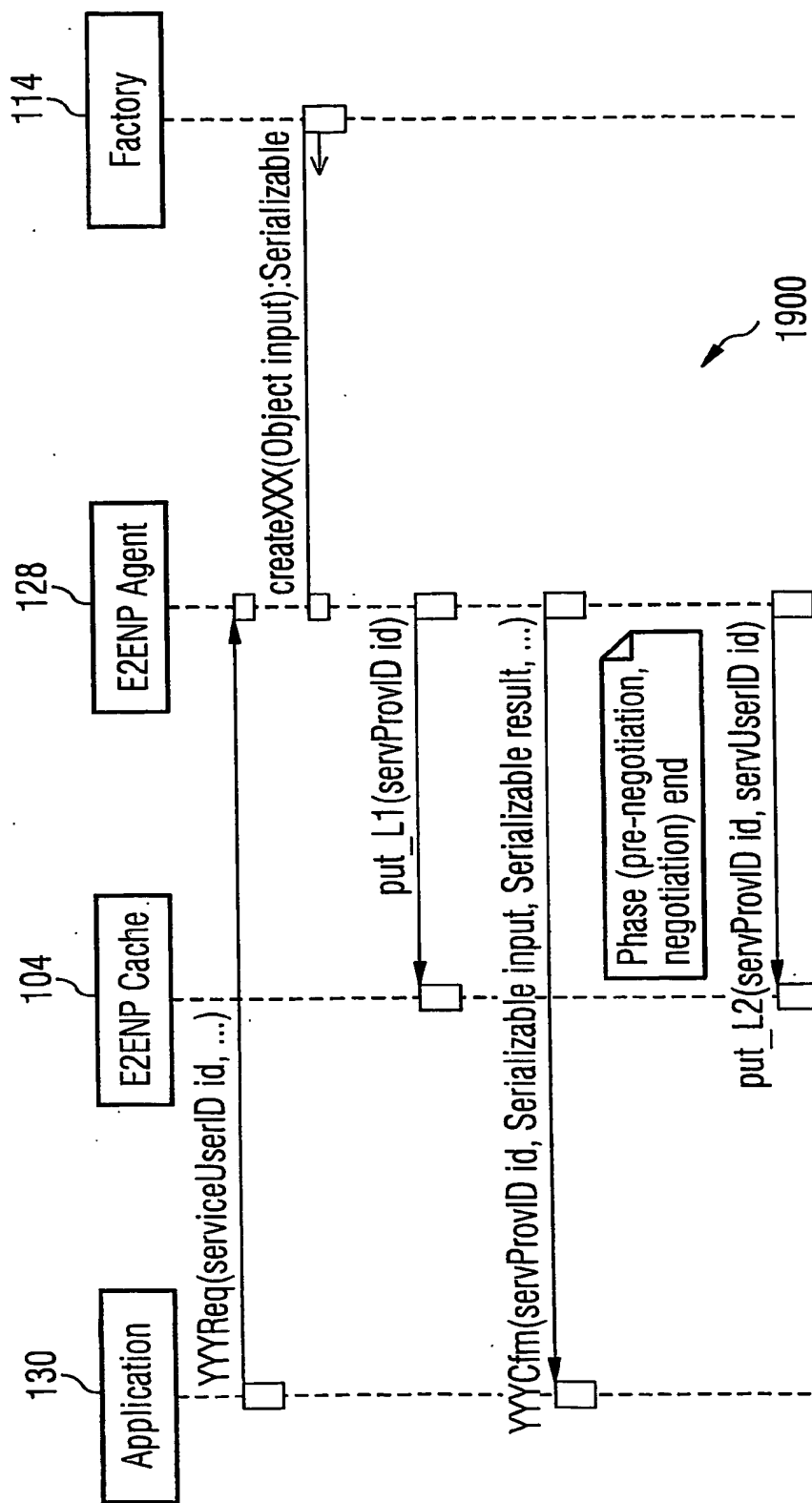


FIG 20

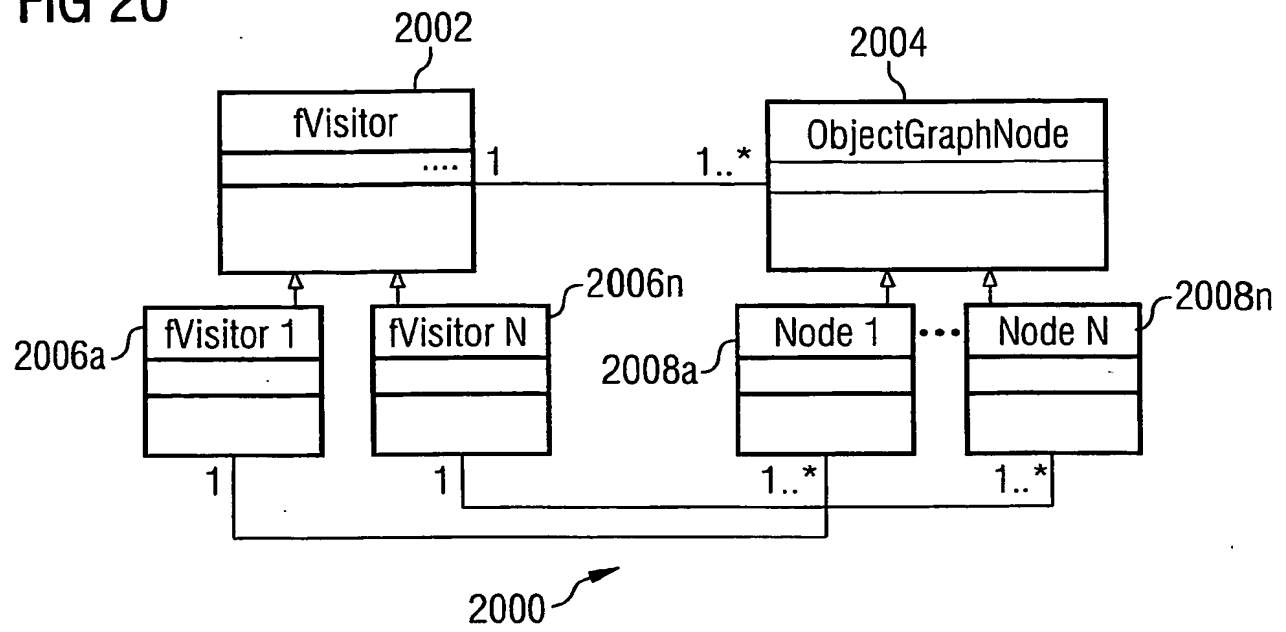


FIG 21

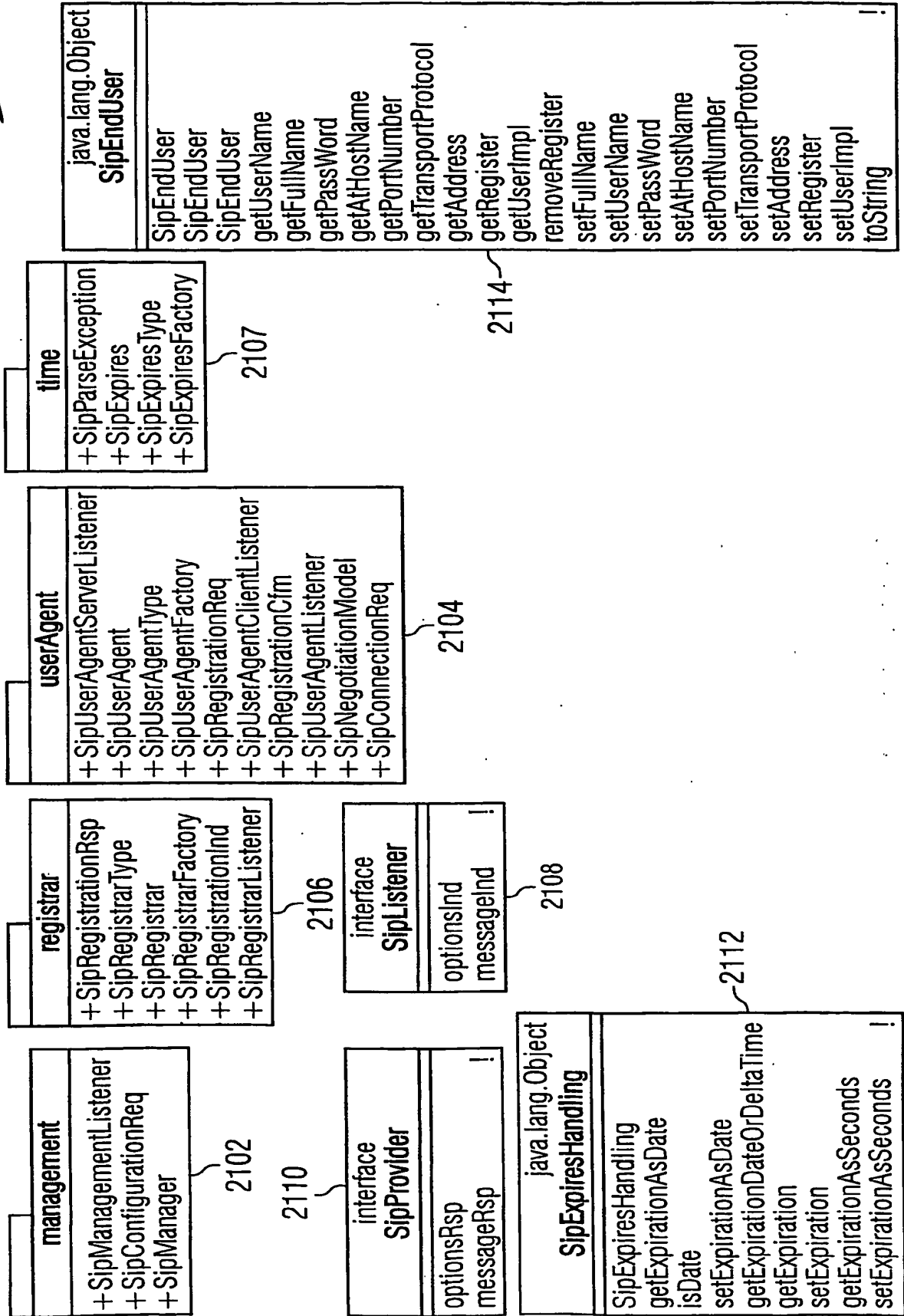


FIG 22

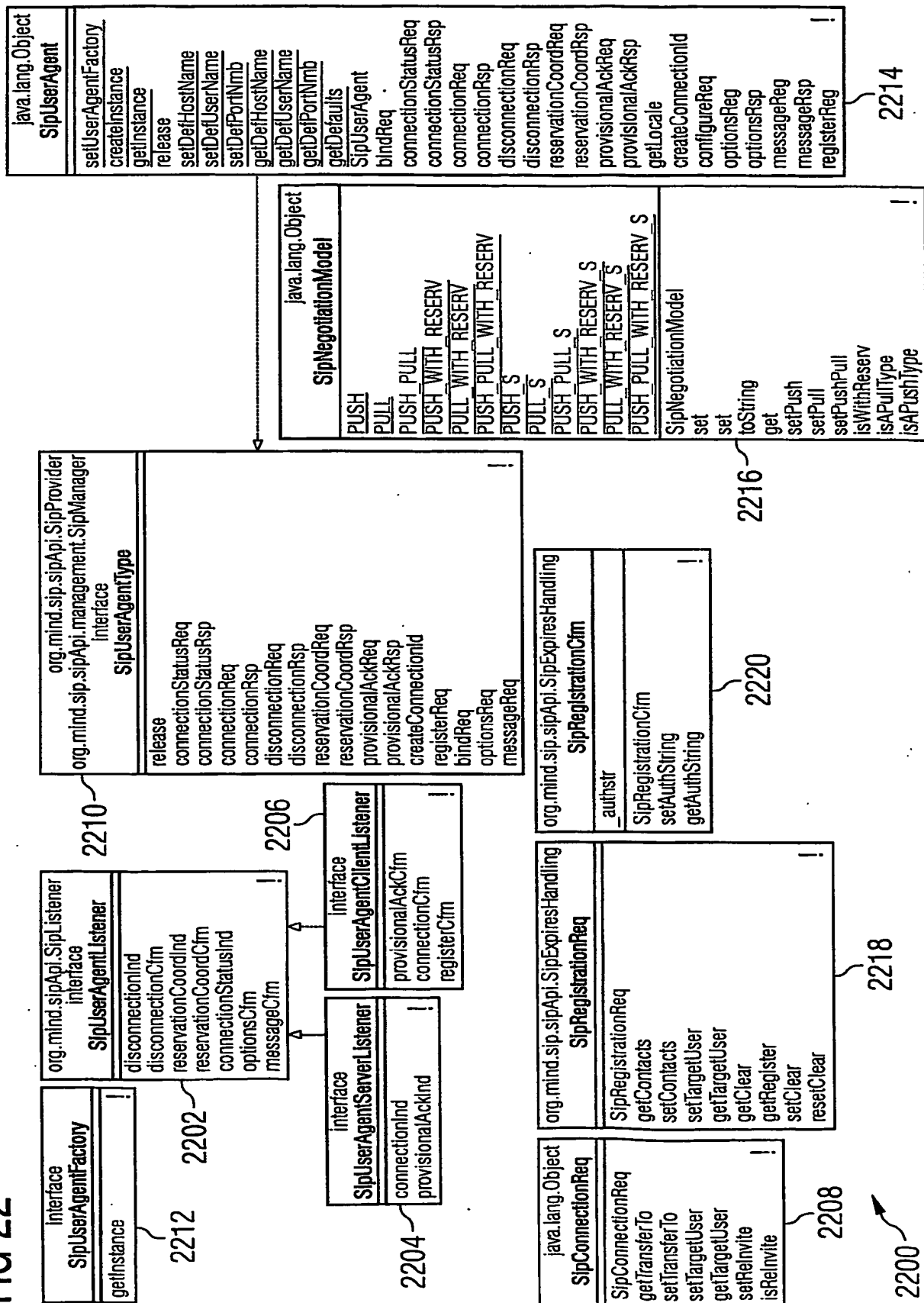
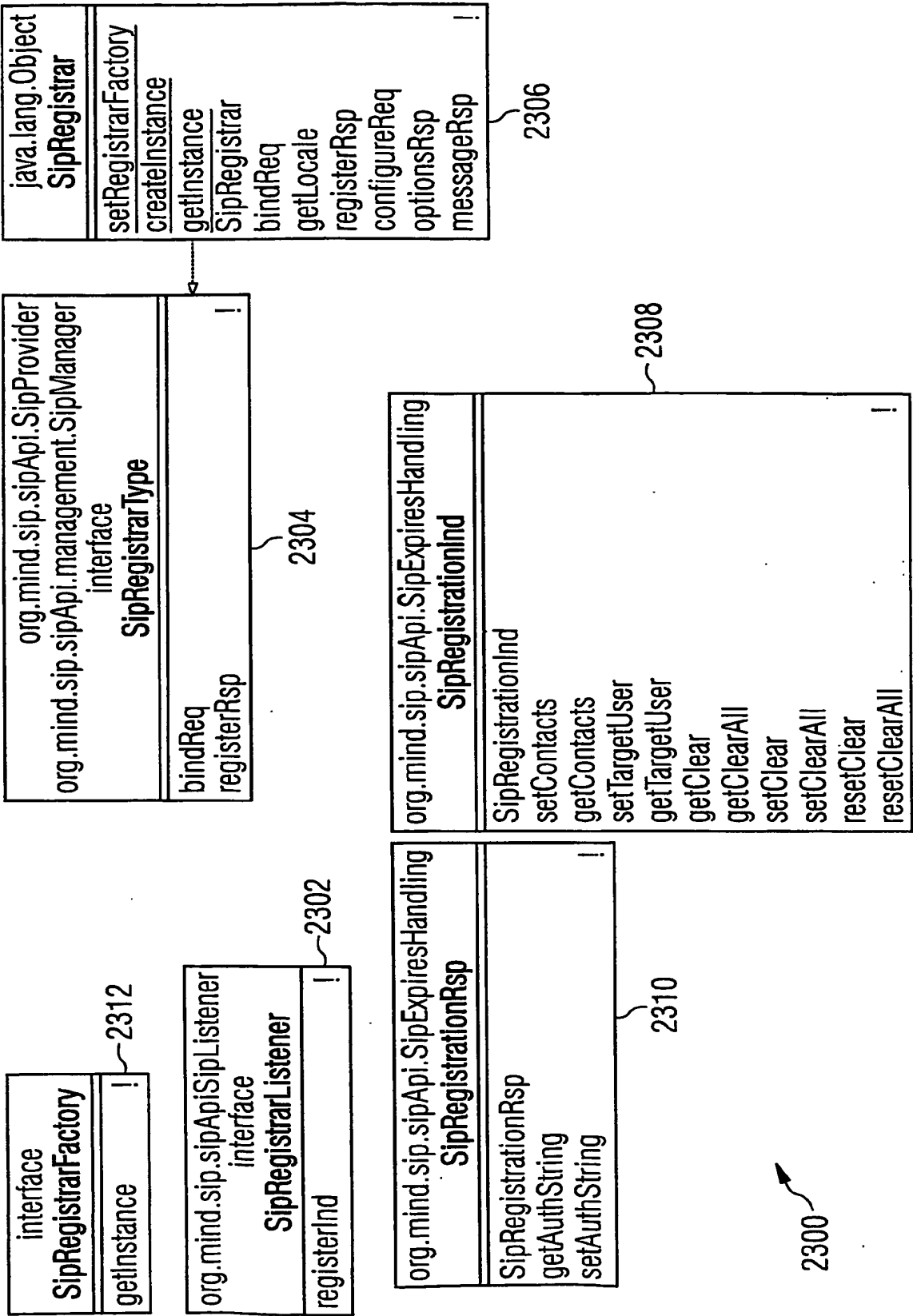


FIG 23



23/58

FIG 24

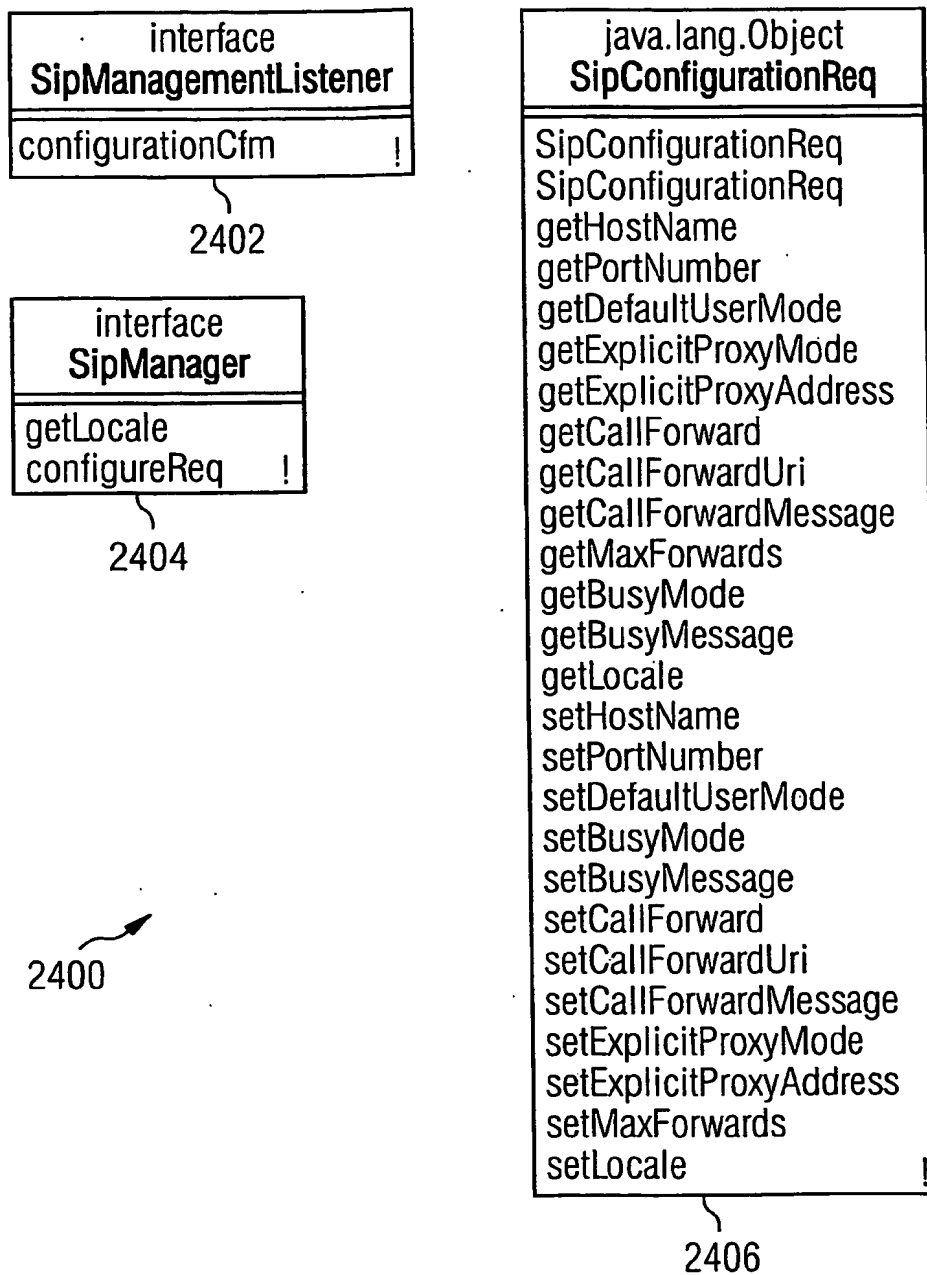


FIG 25

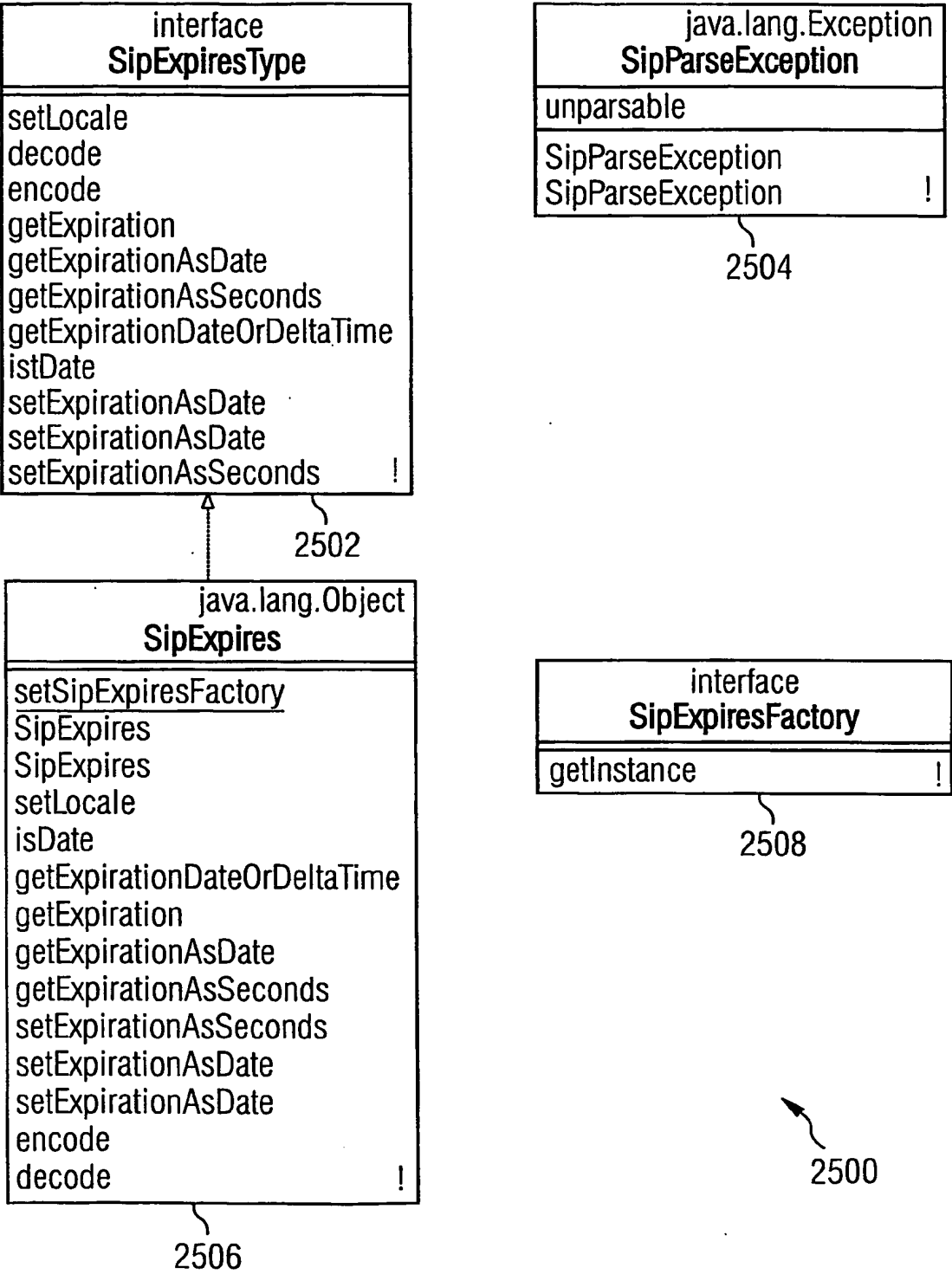


FIG 26

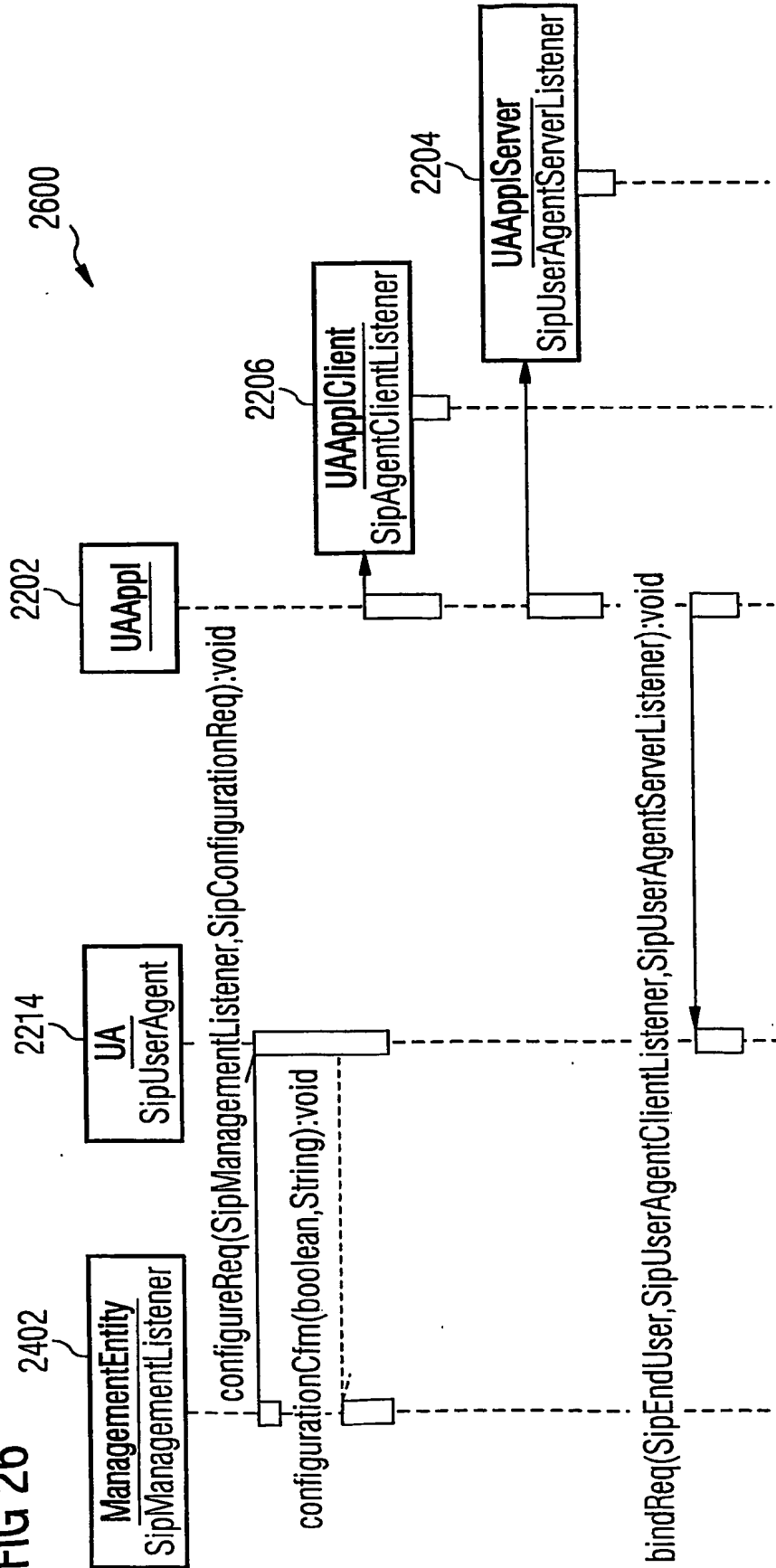


FIG 27
(Part 1)

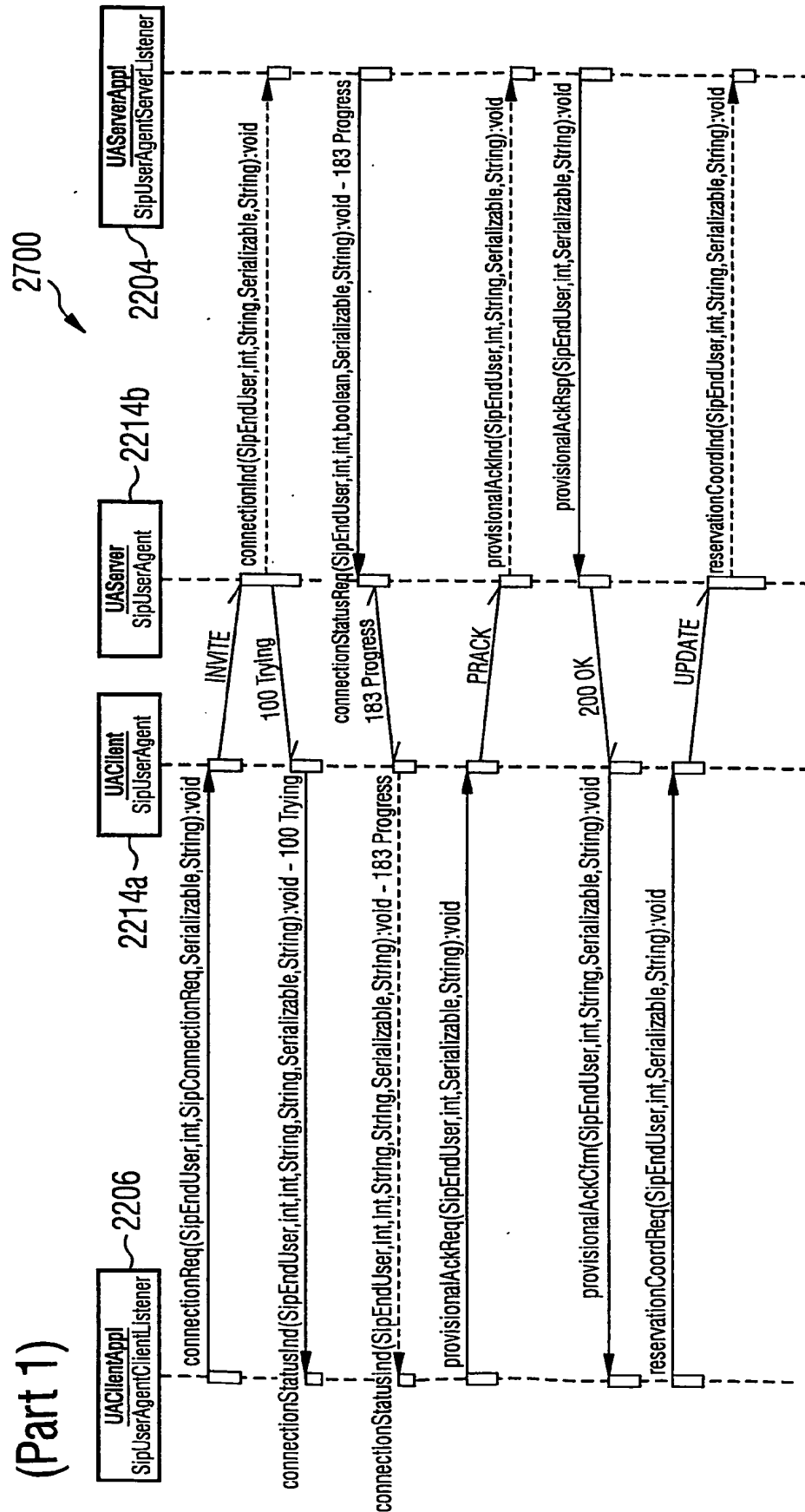
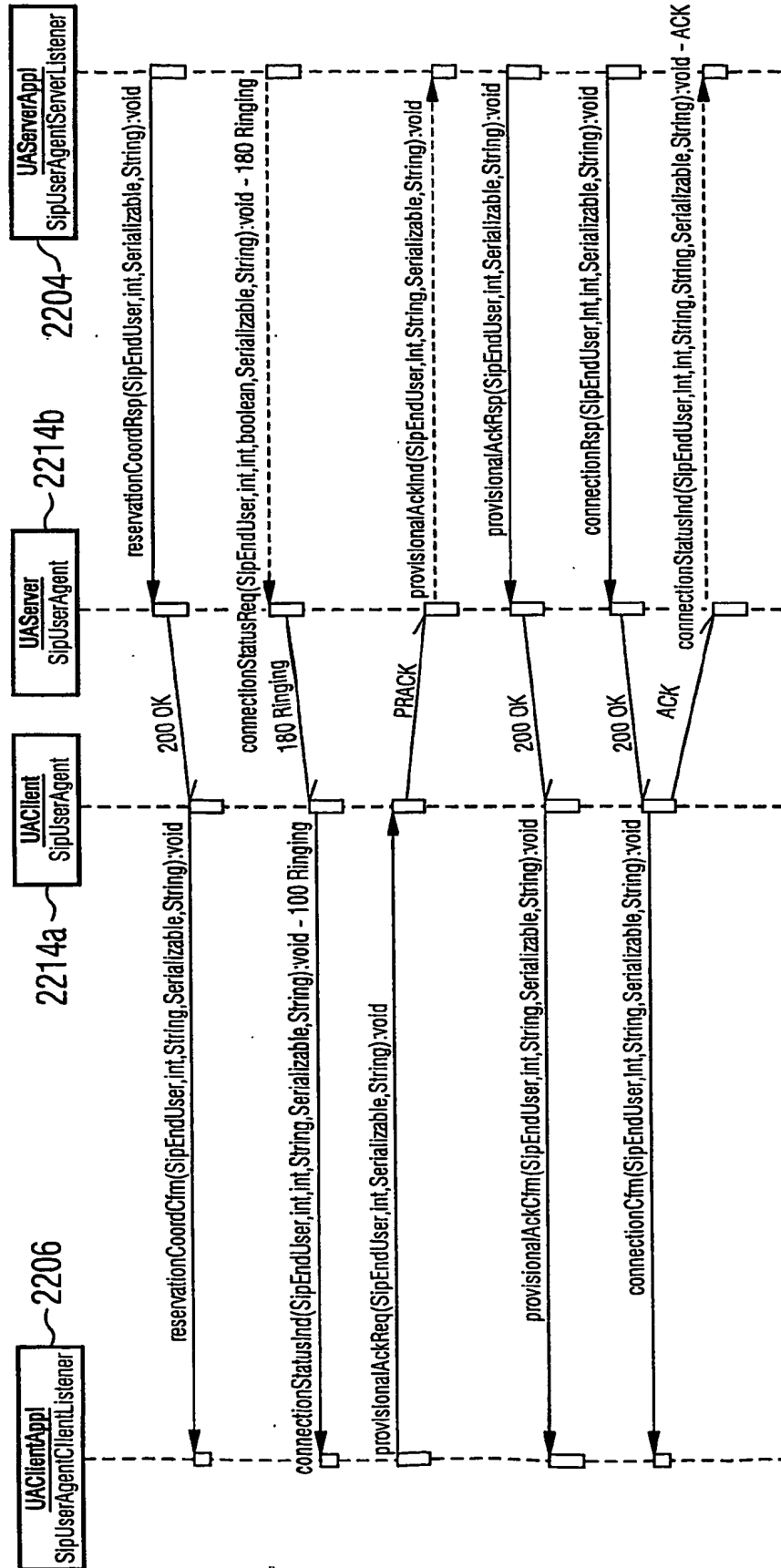
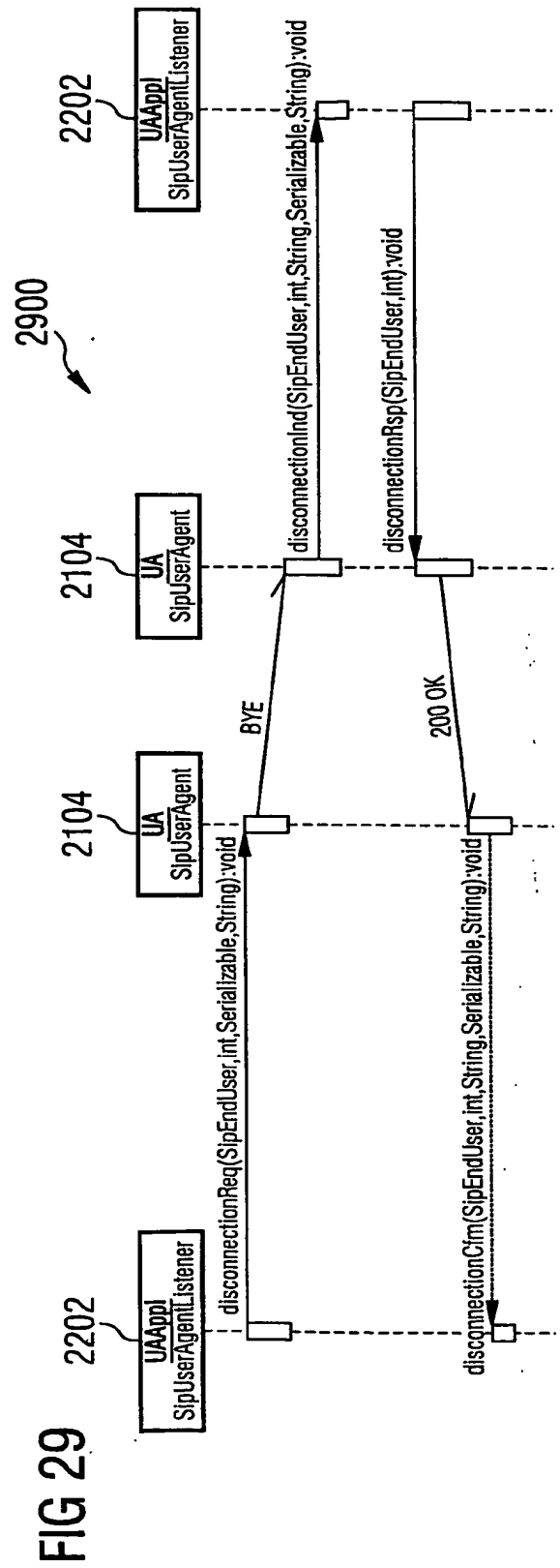
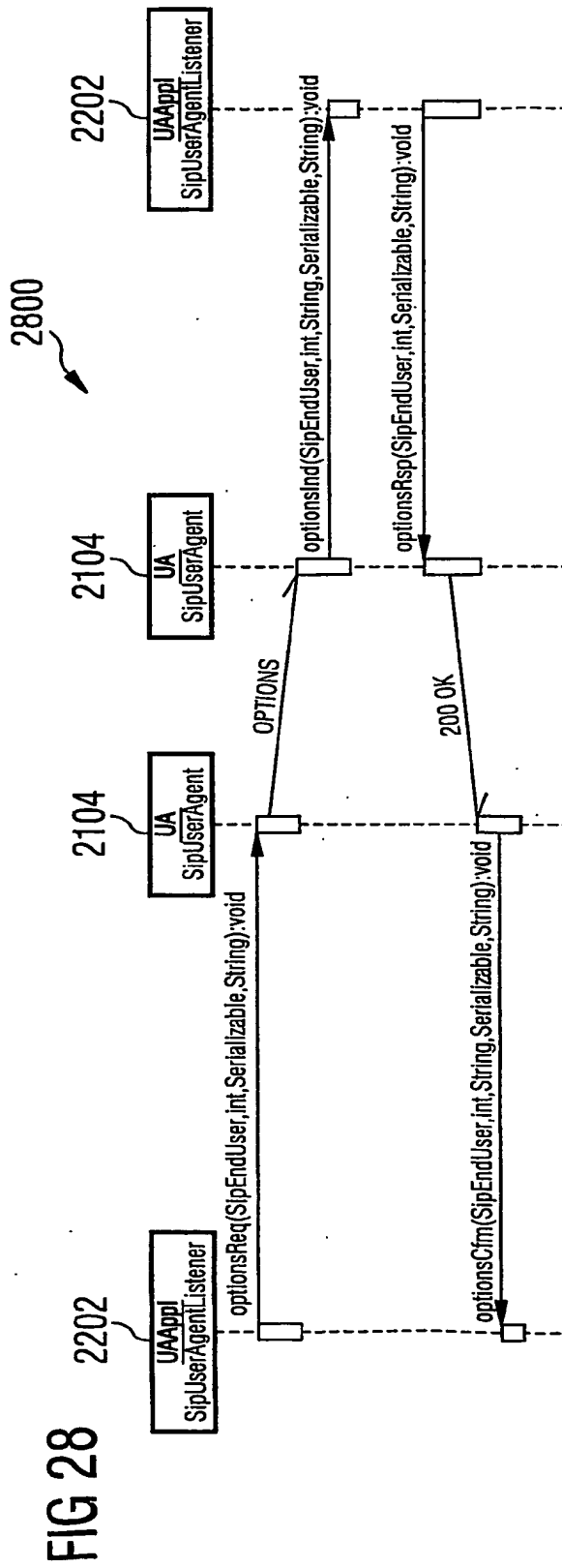


FIG 27
(Part 2)

2700





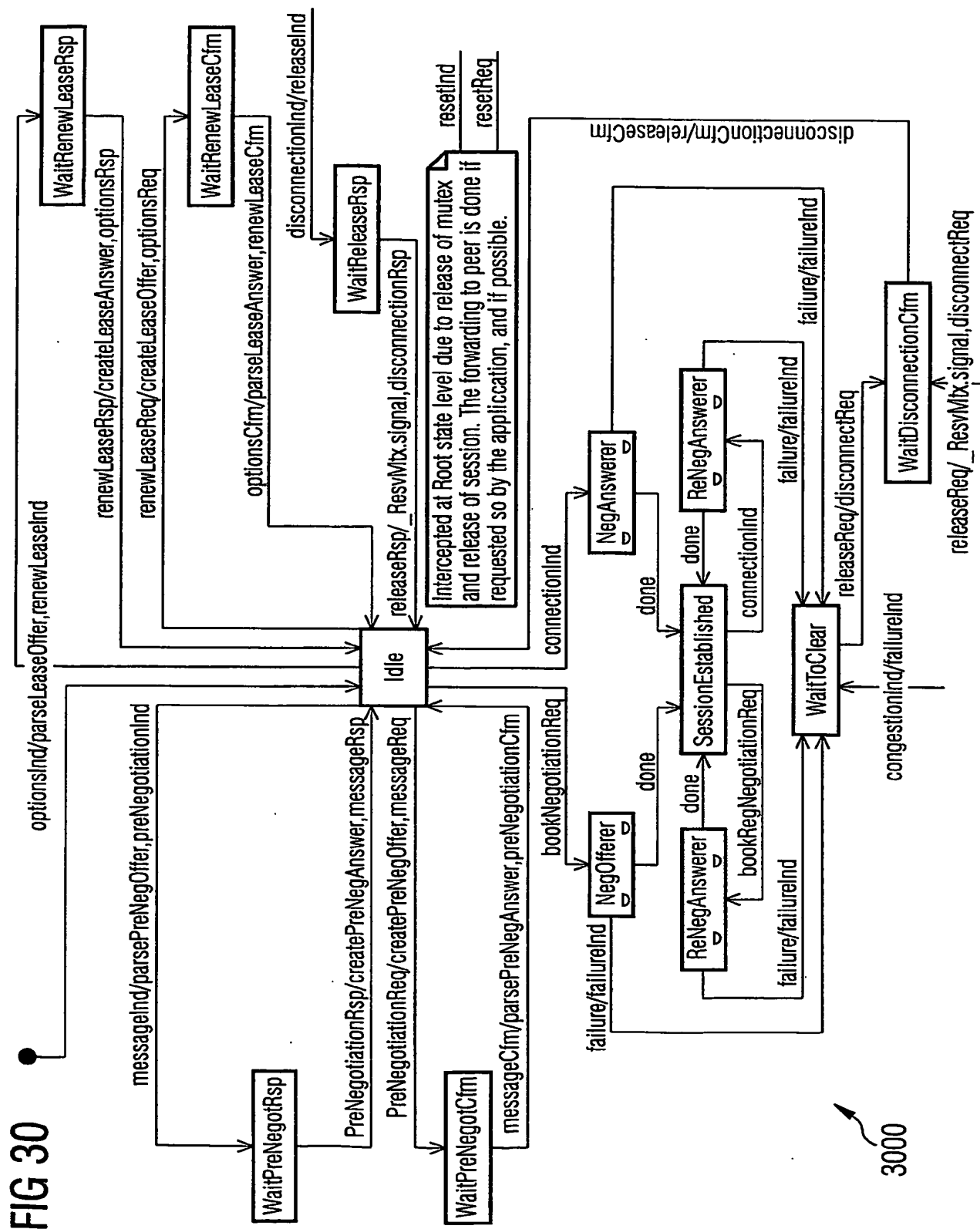


FIG 31

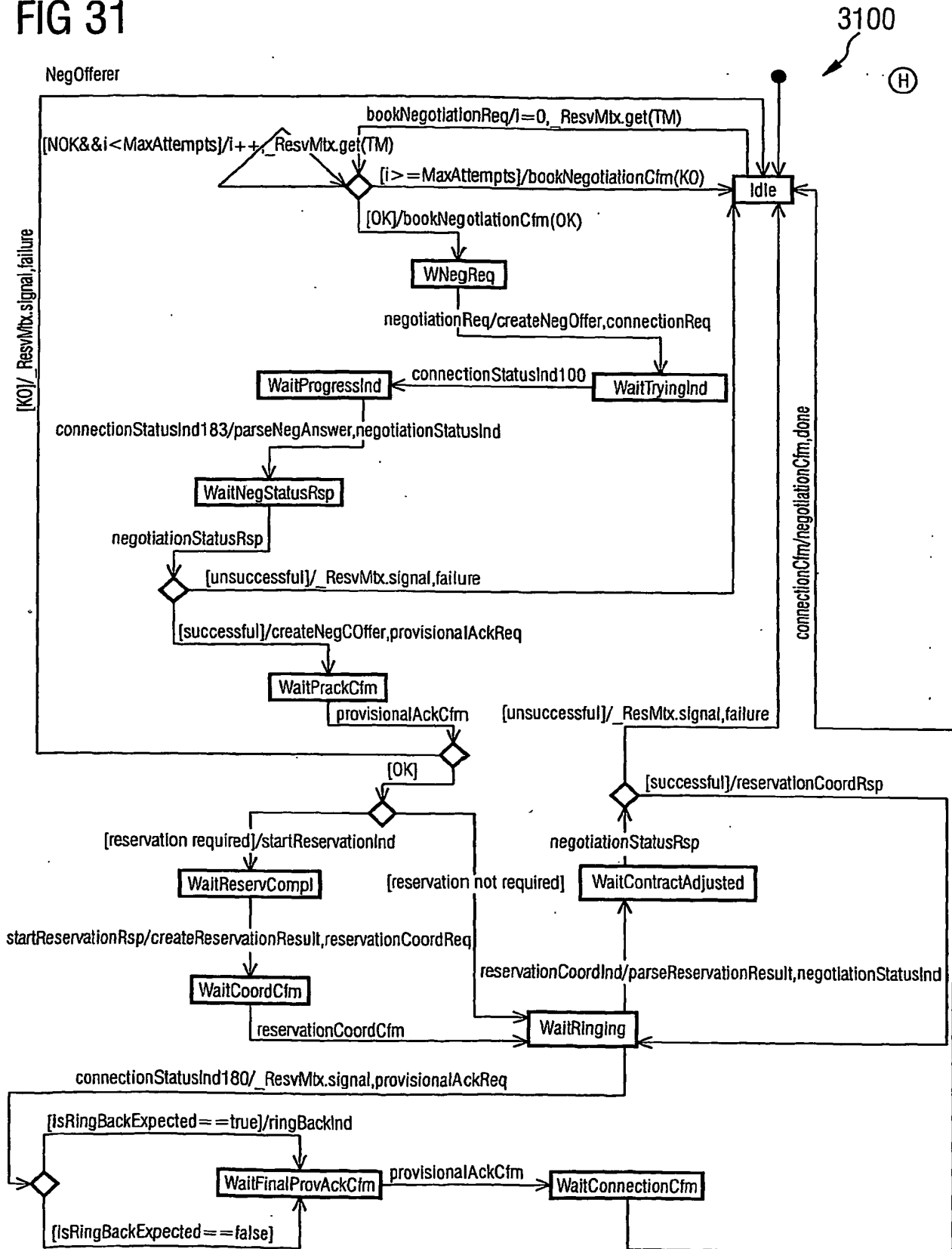


FIG 32

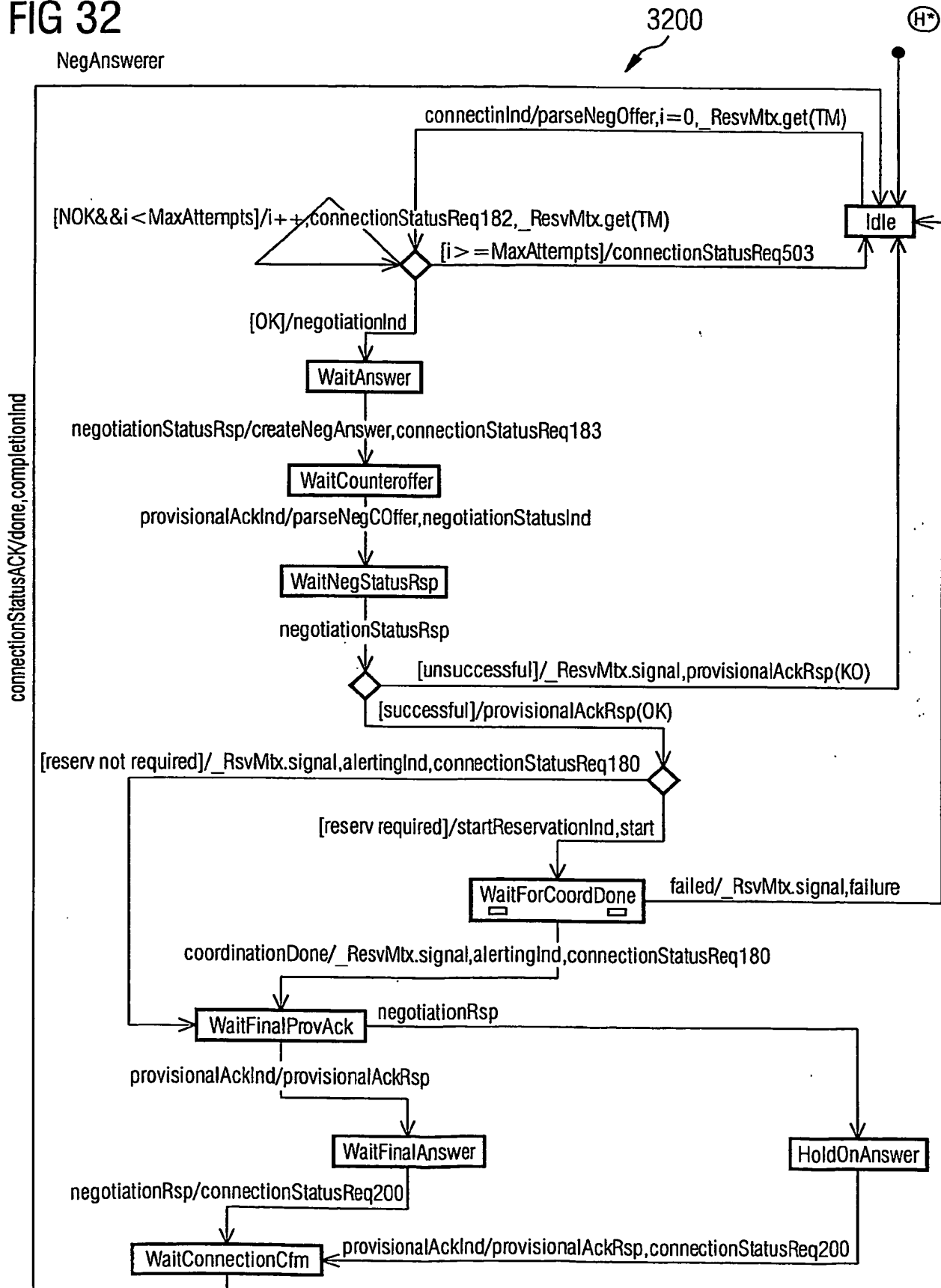


FIG 33

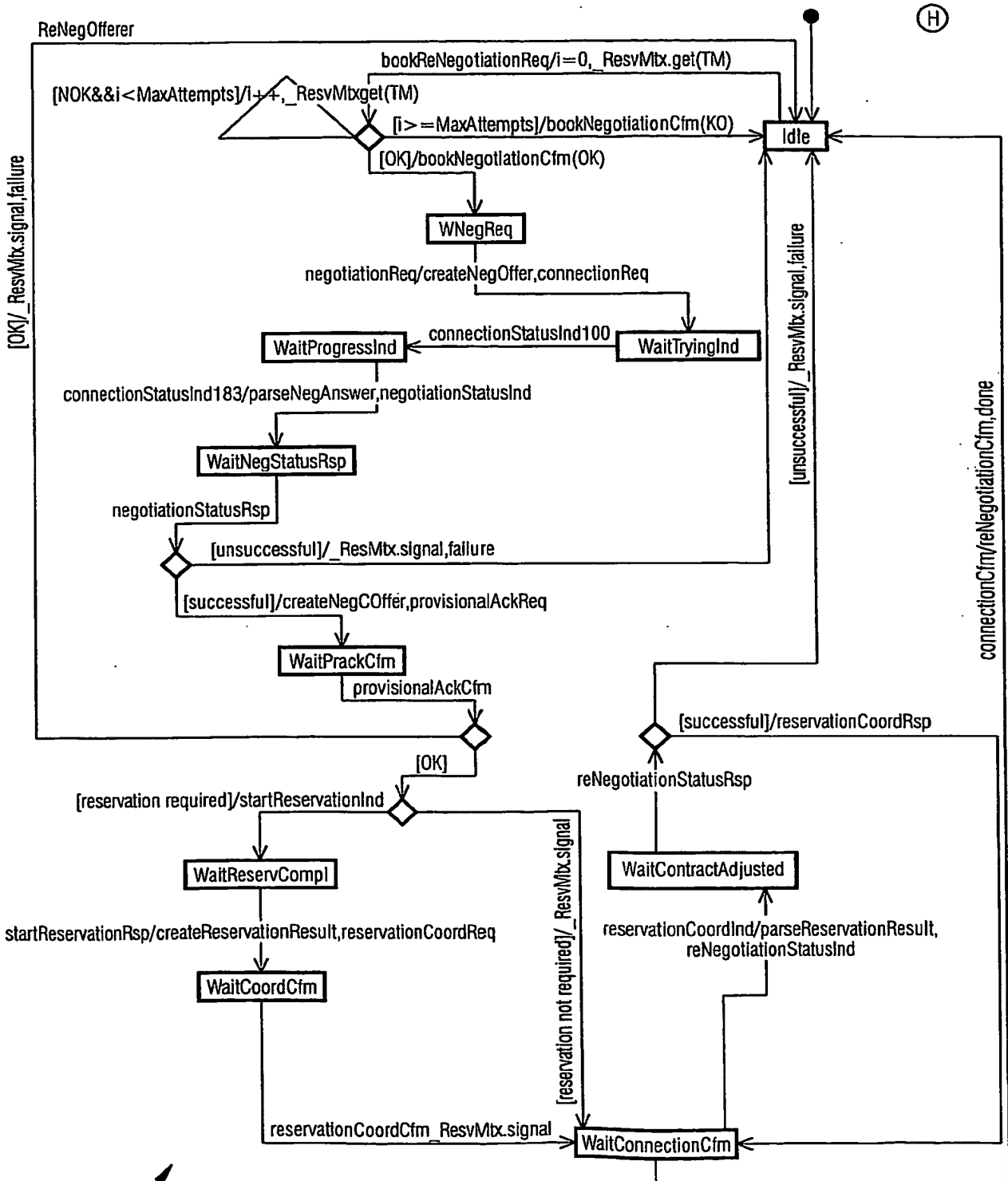


FIG 34

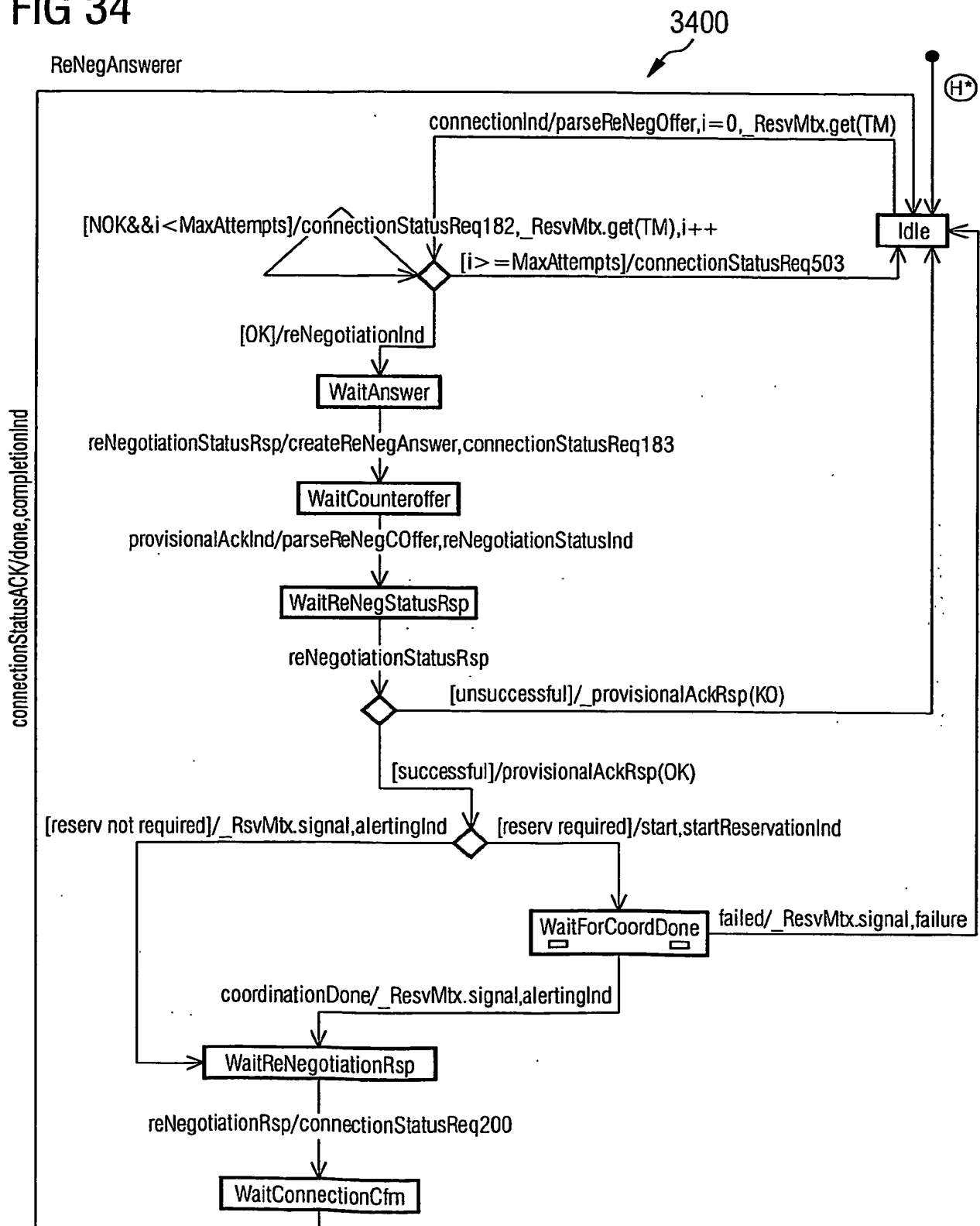
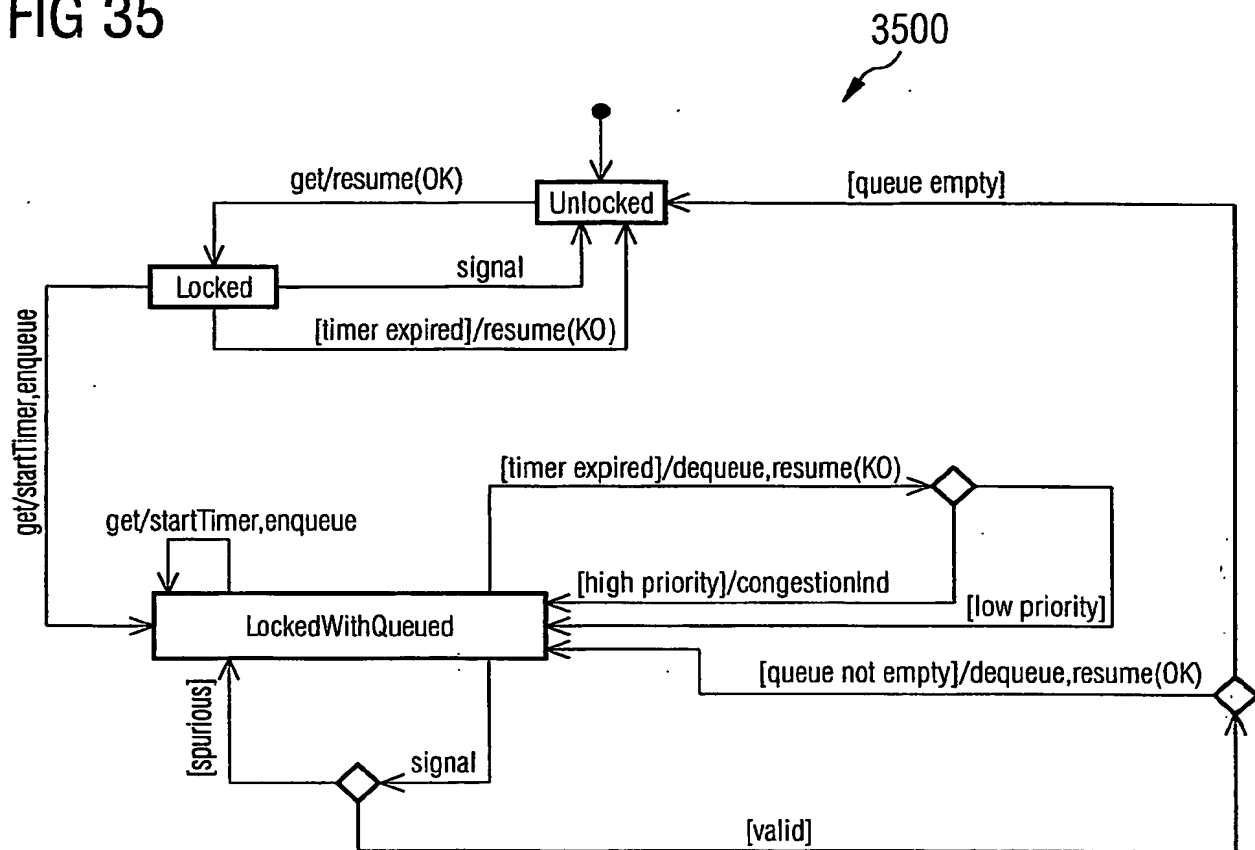


FIG 35



35/58

FIG 36

WaitForCoordDone

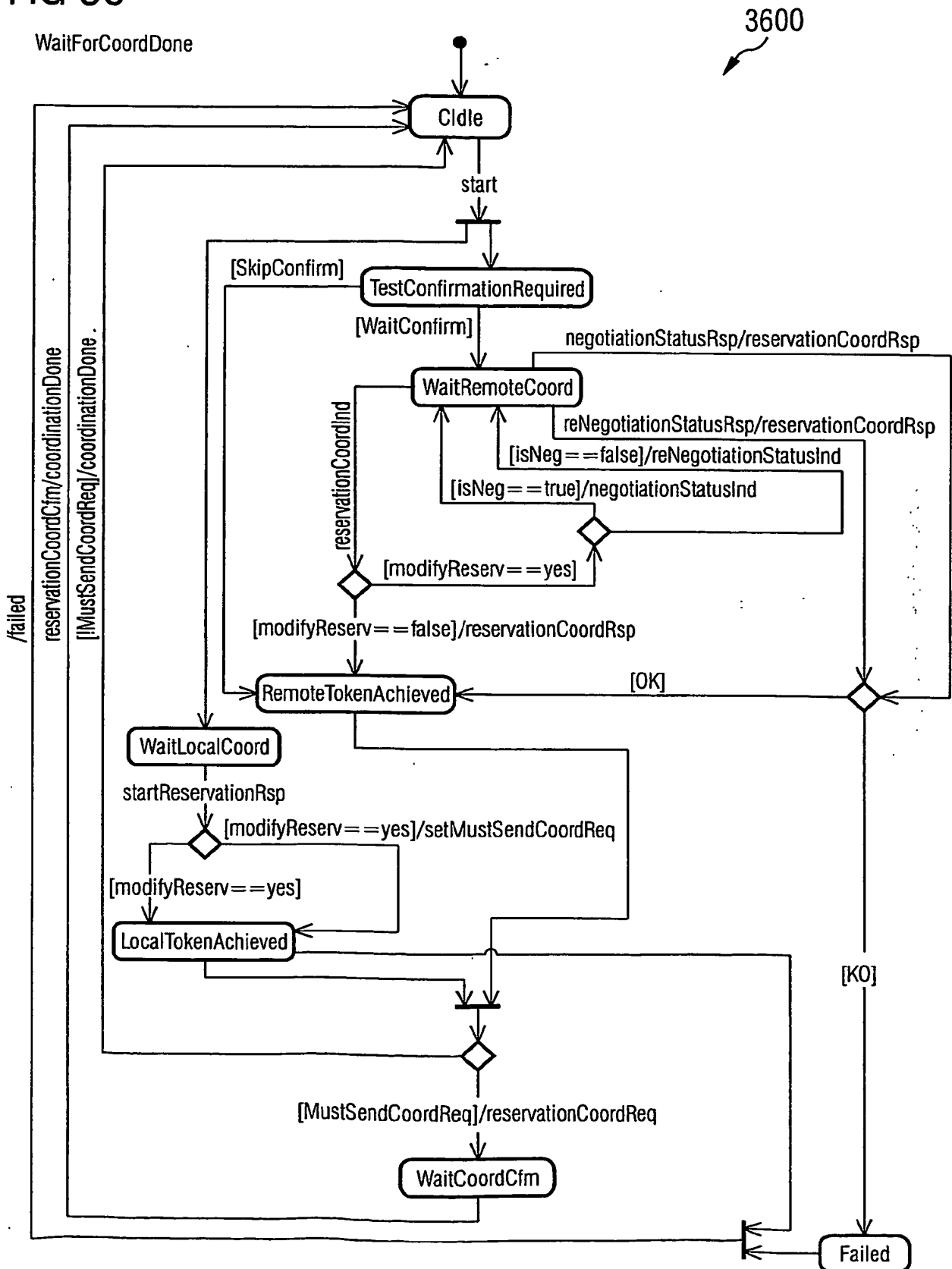


FIG 37

3700

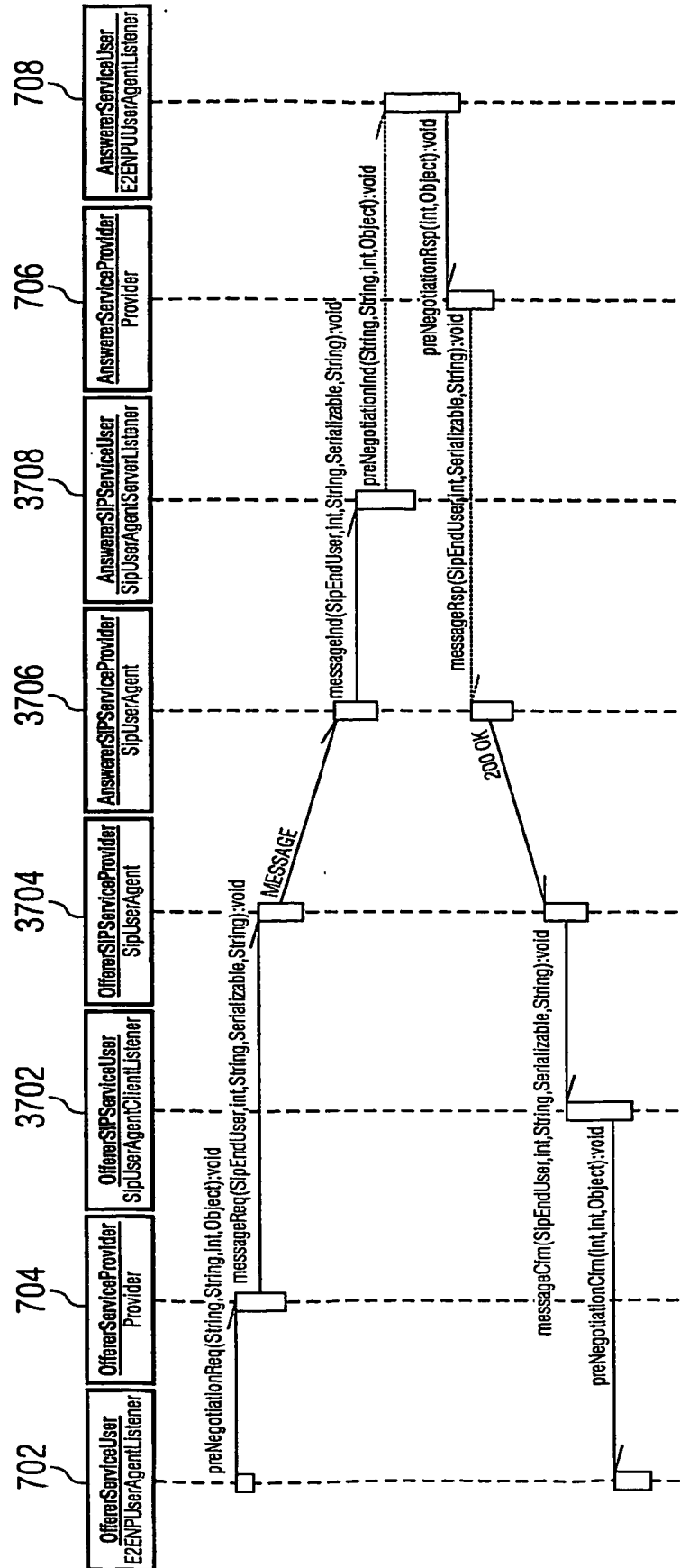


FIG 38
(Part 1)

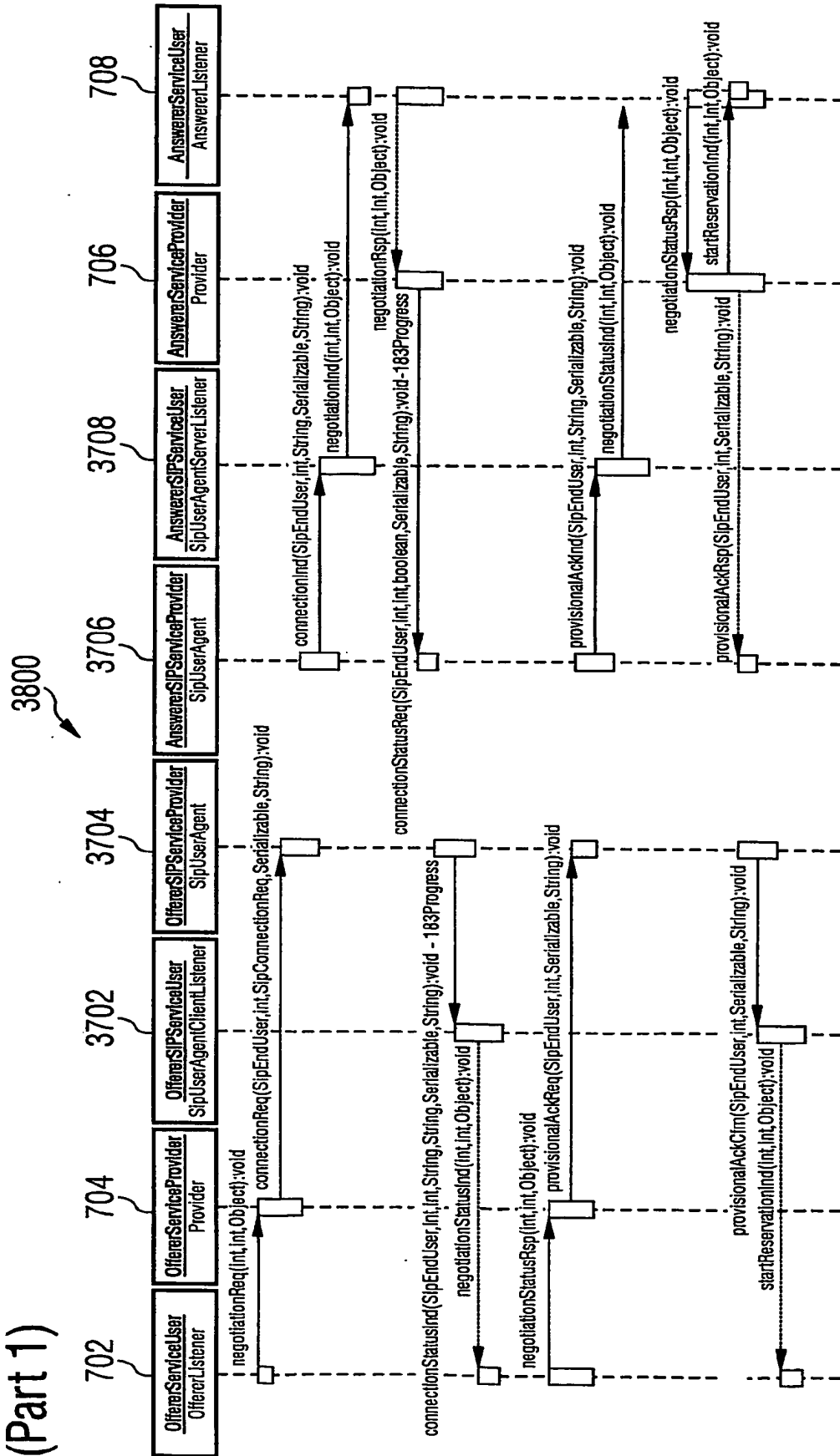
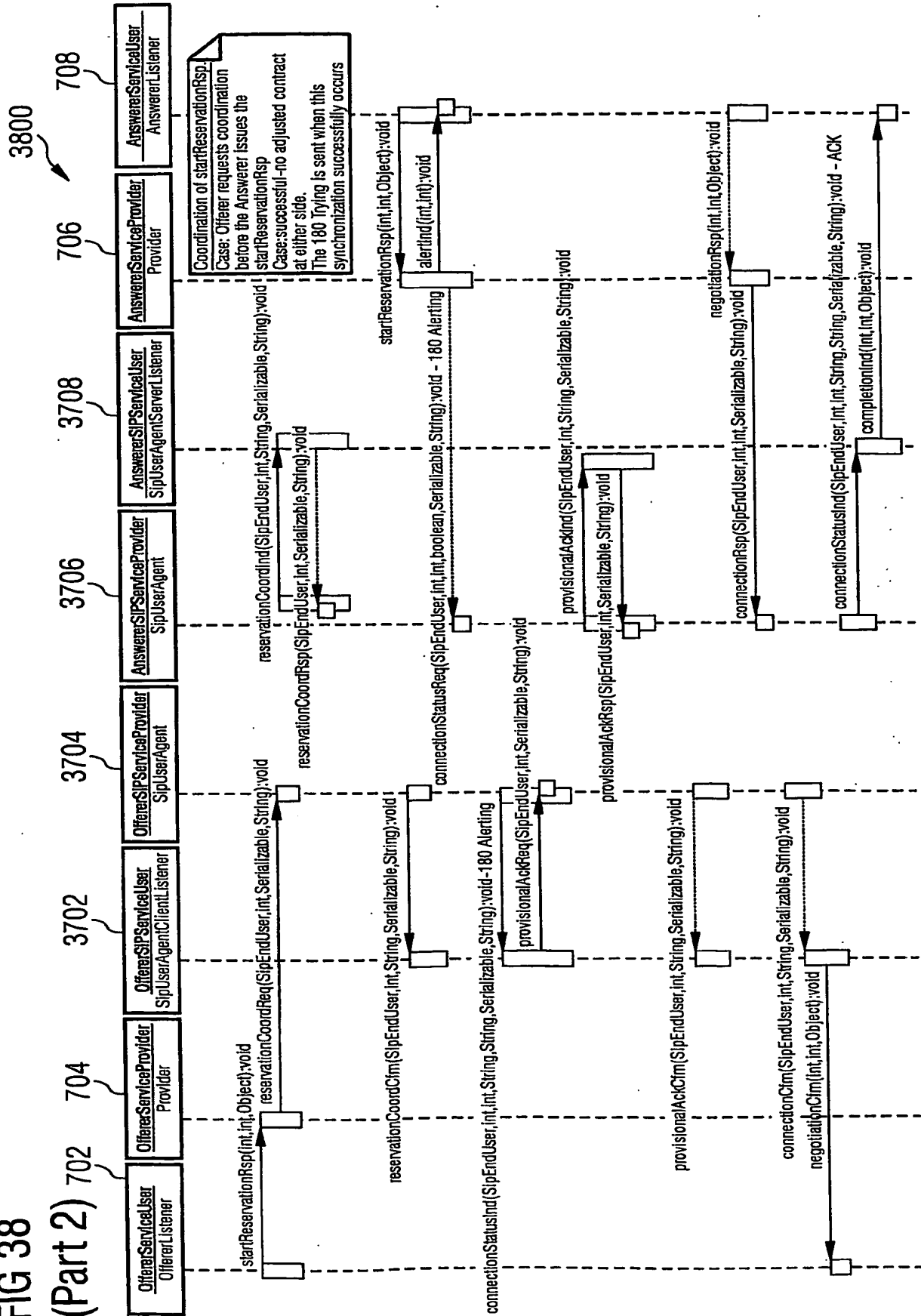


FIG 38

(Part 2)



TAB. 1

3900

```

public void bindReq(OffererServiceUser oSu, AnswererServiceUser a Su, int spld)
    Binds the given Service User's event listeners to the given E2ENP UA.
public void registration Req(int spld, int serviceUserId, java.lang.String user, java.lang.Object info)
    Registers a user with the UA and optionally with a remote registrar.
public void preNegotiationReq(java.lang.String from, java.lang.String to, int serviceUserId,
    java.lang.Object offer)
    Initiates a pre-negotiation phase.
public void preNegotiationRsp(int serviceProviderId, java.lang.Object answer)
    Answers to an incoming request from the peer for pre-negotiation.
public void renewLeaseReq(java.lang.String user, int serviceUserId, java.lang.Object offer)
    Initiates a pre-negotiation information lease refresh.
public void renewLeaseRsp(int serviceProviderId, java.lang.Object answer)
    Answers to an incoming request from the peer for pre-negotiation information lease
    refresh.
public void bookNegotiationReq(java.lang.String user, int serviceUserId, int priority)
    Before initiating the negotiation phase, this primitive allows the Service User to book
    the mutually exclusive use of the resource reservation process, with respect to
    concurrent instances of Service Users.
public void negotiationReq(java.lang.String from, java.lang.String to, int serviceUserId,
    int serviceProviderId, java.lang.Object offer)
    Initiates a negotiation phase.
public void negotiationStatusRsp(int serviceProviderId, java.lang.Object message)
    Generates an intermediate signaling during negotiation in response to a peer's
    signaling.
public void bookReNegotiationReq(int serviceUserId, int priority)
    Before initiating the re-negotiation phase, this primitive allows the Service User to
    book the mutually exclusive use of the resource reservation process,
    with respect to concurrent instances of Service Users. In this case, the
    serviceProviderId must be specified.
public void reNegotiationReq(int serviceProviderId, boolean isPlanned, java.lang.Object offer)
    Initiates a re-negotiation phase.
public void reNegotiationRsp(int serviceProviderId, java.lang.Object answer)
    Answers to an incoming request from the peer for re-negotiation.
public void reNegotiationStatusRsp(int serviceProviderId, java.lang.Object message)
    Generates an intermediate signaling during re-negotiation in response to a peer's
    signaling.
public void startReservationRsp(int serviceProviderId, java.lang.Object result)
    Notifies that the Service User has completed both local and network resource
    reservation.
public void releaseReq(int serviceProviderId, java.lang.Object message)
    Release the given phase (during pre-negotiation and lease renewal) or the overall
    session.
public void releaseRsp(int serviceProviderId, java.lang.Object message)
    Replies to a request from the peer to release the given phase (during pre-negotiation
    and lease renewal) or the overall session.
public void resetReq(int serviceUserId, int serviceProviderId)
    Resets the given session as an emergency procedure.
public void resetRsp(int serviceUserId, int serviceProviderId)
    Replies to a request from the peer to reset the given session as an emergency

```

TAB. 2
(Part 1)4000


```
public void bindCfm (Boolean result)
    Returns the result of the bindReq primitive.
public void registrationCfm (int serviceUserId, java.lang.String user, java.lang.Object info)
    Confirms the registration of the given user with the UA and optionally with an
external
    registrar.
public void preNegotiationInd(java.lang.String from, java.lang.String to, int serviceProviderId,
    java.lang.Object offer)
    Notifies the Service User of a request from the peer to initiate a pre-negotiation
    phase.
public void preNegotiationCfm(int serviceUserId, int serviceProviderId, java.lang.Object answer)
    Notifies the Service User of a reply from the peer concerning the given
    pre-negotiation phase.
public void renewLeaseInd(int serviceUserId, int serviceProviderId, java.lang.Object offer)
    Notifies the Service User of a request from the peer to initiate a pre-negotiation
    information lease refresh.
public void renewLeaseCfm(int serviceUserId, int ServiceProviderId, java.lang.Object answer)
    Notifies the Service User of a reply from the peer concerning the given
    pre-negotiation information lease refresh.
public void negotiationStatusInd(int ServiceUserId, int serviceProviderId, java.lang.Object message)
    Notifies the Service User of an intermediate signaling during negotiation,
    in correspondence to a peer's signaling.
public void bookReNegotiationCfm(int serviceUserId, int serviceUserId, boolean isSuccessful)
    Before initiating the re-negotiation phase, this primitive notifies the Service User that
    the E2ENP UA has successfully booked the mutually exclusive use of the resource
    reservation process, with respect to concurrent instances of Service Users.
public void reNegotiationInd(int serviceUserId, int serviceProviderId, boolean isPlanned,
    java.lang.Object offer)
    Notifies the Service User of a request from the peer to initiate a re-negotiation phase.
public void reNegotiationCfm(int serviceUserId, int serviceProviderId, java.lang.Object answer)
    Notifies the Service User of a reply from the peer concerning the given re-
negotiaton.
public void reNegotiationStatusInd(int serviceUserId, int serviceProviderId, java.lang.Object
    message)
    Notifies the Service User of an intermediate signaling during re-negotiation, in
    correspondence to a peer's signaling.
public void startReservationInd(int serviceUserId, int serviceProviderId, java.lang.Object result)
```


TAB. 2
(Part 2)4000


```
public void releaseCfm(int serviceUserId, int serviceProviderId, java.lang.Object message)
    Notifies the Service User of a reply from the peer concerning the release of the given
    phase (during pre-negotiation and lease renewal) or the overall session.
public void alertInd(int serviceUserId, int serviceProviderId)
    Notifies the Service User that a remote peer has called in.
public void ringBackInd(int serviceUserId, int serviceProviderId)
    Notifies the Service User that the remote peer has been alerted.
public void congestionInd(int serviceUserId, int serviceProviderId)
    Notifies the given Service User instance that other higher priority instances are
    waiting for the mutex to be released. The Service User shall pre-empt by rolling back
    to the state before the negotiation/re-negotiation started, and invoke the release
    primitive.
public void failureInd(int serviceUserId, int serviceProviderId)
    Notifies the Service User that an E2ENP UA internal error occurred and that the
    Service User shall invoke the release primitive.
public void abortInd(int serviceUserId, int serviceProviderId, int reason)
    Notifies the Service User that a SIP error occurred and that the Service User shall
    simply consider the current operation as aborted. A reason is passed as well,
    indicating one of the possible sources of error:
        0 - redirection,
        1 - client error,
        2 - server error,
        3 - network failure.
public void resetInd(int serviceUserId, int serviceProviderId)
    Notifies the Service User of a request from the peer concerning the reset of the
    session.
public void resetCfm(int serviceUserId, int serviceProviderId)
```

TAB. 3

4100

public void bookNegotiationCfm(int serviceUserId, int serviceUserId, boolean isSuccessfull)
Before initiating the negotiation phase, this primitive notifies the Service User that the E2ENP UA has successfully booked the mutually exclusive use of the resource reservation process, with respect to concurrent instances of Service Users. This primitive is specific of the offerer side, whereas the homonymous **bookReNegotiationCfm** may be invoked either at the offerer or at the answerer side.

public void negotiationCfm(int serviceUserId, int serviceProviderId, java.lang.Object Answer)
Notifies the Service User of a reply from the peer concerning the given negotiation.

TAB. 4

4200

public void negotiationInd(java.lang.String from, java.lang.String to, int serviceUserId, int serviceProviderId, java.lang.Object offer)
Notifies the Service User of a request from the peer to initiate a negotiation phase.

public void completionInd(nt serviceUserId, int serviceProviderId, Object message)
Notifies the Service User that the negotiation/setup phase has been completed successfully.

TAB. 5

4300


public LevelOneEntry	LevelOneCache.createEntry(CoreSession session,int serviceProviderId) Creates a level 1 entry.
public LevelOneEntry	LevelOneCache.getEntry(int serviceProviderId) Gets a level 1 entry by using the service provider ID as primary key.
public LevelOneEntry	LevelOneCache.findEntry(int serviceUserId) Gets a level 1 entry using the service user ID as secondary key.
public LevelOneEntry	LevelOneCache.findEntry(java.lang.String from,java.lang.String to) Gets a level 1 entry using the(From,To)SIP address couple as secondary key.
public java.util.Vector	LevelOneCache.findEntryByExtSessionId(java.lang.String sessionId) Gets the level 1 entries by using the E2ENP Session ID as secondary key.
public void	LevelOneCache.removeEntry(LevelOneEntry entry) Deletes the given entry from the cache.
public void	LevelOneCache.clear() Removes all the entries.
public void	LevelOneEntry.putSessionID(java.lang.String sessionId) Puts the E2ENP Session ID in the entry associated with the given service provider ID.
public java.lang.String	LevelOneEntry.getSessionID() Gets the E2ENP Session ID associated with the given service provider ID.
public void	LevelOneEntry.putServiceUserID(int serviceUserId) Puts the service user ID in the entry associated with the given service provider
public int	LevelOneEntry.getServiceUserID() Gets a service user ID.
public int	LevelOneEntry.getServiceProviderID() Gets a service provider ID.
public void	LevelOneEntry.putFromAddress(java.lang.String from) Puts the SIP From Address associated with the given service provider ID.
public java.lang.String	LevelOneEntry.getFromAddress() Gets the SIP From Address associated with the given service provider ID.
public void	LevelOneEntry.putToAddress(java.lang.String to) Puts the SIP To Address associated with the given service provider ID.
public java.lang.String	LevelOneEntry.getToAddress() Gets the SIP To Address associated with the given service provider ID.

TAB. 6

4400


public LevelTwoEntry	LevelTwoCache.createEntry(int serviceProviderId) Creates a level 2 entry.
public LevelTwoEntry	LevelTwoCache.findEntry(java.lang.String sessionId) Gets all the level 2 entries by using the E2ENP Session ID as primary key.
public void	LevelTwoCache.removeEntry(LevelTwoEntry entry) Deletes the given entry from the cache.
public void	LevelTwoCache.clear() Removes all the entries.
public void	LevelTwoEntry.putSessionID(java.lang.String sessionId) Puts the E2ENP Session ID in the entry associated with the given service provider ID.
public java.lang.String	LevelTwoEntry.getSessionID() Gets the E2ENP Session ID associated with the given service provider ID.
public void	LevelTwoEntry.putServiceUserID(int serviceUserId) Puts the service user ID in the entry associated with the given service provider
public int	LevelTwoEntry.getServiceUserID() Gets a service user ID.

TAB. 7 (Part 1)

4500


java.lang.String	getType() Returns an identifier to uniquely specify the type of this parser.
boolean	isConfirmExpected(java.io.Serializable in) Checks all media stream definitions for preconditions (PC), and, if at least one such definition has a PC that the offerer is going to carry out reservation, and thus, going to send a confirmation to the answerer when done.
boolean	isKeepResultsCached(java.io.Serializable in) Extracts information about whether the results of a negotiation of re-negotiations should be cached or not.
boolean	isRingBackExpected(java.io.Serializable in) Checks all media stream definitions and returns true, if any of the streams is going to be generated by the answerer.
java.lang.Object	parseFinalResponse(java.io.Serializable in) Parses a final result message and returns the corresponding status.
java.lang.Object	parseLeaseAnswer(java.io.Serializable in) Parses a transport representation of a renew lease answer and returns the corresponding object representation.
java.lang.Object	parseLeaseOffer(java.io.Serializable in) Parses a transport representation of a renew lease offer and returns the corresponding object representation.
long	parseLeaseTime(java.io.Serializable in) Parses a transport representation of a lease answer/offer and returns the corresponding lease time as a basic long value.
java.util.Vector	parseListOfUsedSessionIds(java.io.Serializable in) Parses the given input selectively to extract the external representation of the list of referenced E2ENP session identifier.
java.lang.Object	parseMessage(java.io.Serializable in) Parses content which is mapped over various e2enp primitives/SIP messages.
java.lang.Object	parseNegAnswer(java.io.Serializable in) Parses a transport representation of a negotiation answer and returns the corresponding object representation.
java.lang.Object	parseNegCOffer(java.io.Serializable in) Parses a transport representation of a negotiation counter-offer and returns the corresponding object representation.
java.lang.Object	parseNegOffer(java.io.Serializable in) Parses a transport representation of a negotiation offer and returns the corresponding object representation.

TAB. 7 (Part 2)

4500


java.lang.Object	<u>parsePreNegAnswer</u> (java.io.Serializable in) Parses a transport representation of a pre-negotiation answer and returns the corresponding object representation.
java.lang.Object	<u>parsePreNegOffer</u> (java.io.Serializable in) Parses a transport representation of a pre-negotiation offer and returns the corresponding object representation.
java.lang.Object	<u>parseReNegAnswer</u> (java.io.Serializable in) Parses a transport representation of a re-negotiation answer and returns the corresponding object representation.
java.lang.Object	<u>parseReNegCOffer</u> (java.io.Serializable in) Parses a transport representation of a re-negotiation counter-offer and returns the corresponding object representation.
java.lang.Object	<u>parseReNegOffer</u> (java.io.Serializable in) Parses a transport representation of a re-negotiation offer and returns the corresponding object representation.
java.lang.Object	<u>parseReservationResult</u> (java.io.Serializable in) Parses a transport representation of a reservation result message and returns the corresponding object representation.
java.lang.String	<u>parseSessionId</u> (java.io.Serializable in) Parses the given input selectively to extract the external representation of the E2ENP session identifier.

TAB. 8

4600


Parser	<u>createParser</u> (java.lang.String protocolID) Creates a parser instance for the given protocol.
void	<u>registerParser</u> (java.lang.String protocolID, java.lang.String parserClassName) Registers a parser implementation for the given protocol.

TAB. 9

4700


java.io.Serializable	createFinalResponse (java.lang.Object status) Creates the transport representation for a final response message.
java.io.Serializable	createLeaseAnswer (java.lang.Object in) Creates the transport representation for a renew lease answer.
java.io.Serializable	createLeaseOffer (java.lang.Object in) Creates the transport representation for a renew lease offer.
java.io.Serializable	createMessage (java.lang.Object in) Creates serializable representation of content which is mapped over various
java.io.Serializable	createNegAnswer (java.lang.Object in) Creates the transport representation for a negotiation answer.
java.io.Serializable	createNegCOffer (java.lang.Object in) Creates the transport representation for a negotiation counteroffer.
java.io.Serializable	createNegOffer (java.lang.Object in) Creates the transport representation for a negotiation offer.
java.io.Serializable	createPreNegAnswer (java.lang.Object in) Creates the transport representation for a pre-negotiation answer.
java.io.Serializable	createPreNegOffer (java.lang.Object in) Creates the transport representation for a pre-negotiation offer.
java.io.Serializable	createReNegAnswer (java.lang.Object in) Creates the transport representation for a re-negotiation answer.
java.io.Serializable	createReNegCOffer (java.lang.Object in) Creates the transport representation for a re-negotiation counteroffer.
java.io.Serializable	createReNegOffer (java.lang.Object in) Creates the transport representation for a re-negotiation offer.
java.io.Serializable	createReservationResult (java.lang.Object in) Creates the transport representation for a reservation result message.
long	getLeaseTime (java.lang.Object in) Extracts the lease time stored in the internal representation represented by the input parameter 'in'.
java.util.Vector	getListOfUsedSessionIds (java.lang.Object in) Extracts the list of used session IDs stored in the internal representation represented by the input parameter 'in'.
java.lang.String	getSessionId (java.lang.Object in) Extracts the session ID stored in the internal representation represented by the input parameter 'in'.
java.lang.String	getType () Returns an identifier to uniquely specify the type of this factory.
boolean	isKeepResultsCached (java.lang.Object in) Extracts information about whether the results of a negotiation or of re-negotiations

TAB. 10

4800


Factory	createFactory (java.lang.String protocolID)
void	registerFactory (java.lang.String protocolID, java.lang.String factoryClassName) Registers a factory class for the given protocol.

TAB. 11

4900


abstract <u>Factory</u>	<u>createFactory</u> (java.lang.String protocolID) Creates a factory instance for the given protocol.
abstract <u>Parser</u>	<u>createParser</u> (java.lang.String protocolID) Creates a parser instance for the given protocol.
static <u>E2ENPContentHandlerFactory</u>	<u>getInstance</u> () Provides access to the singleton instance of the E2ENPContentHandlerFactory.
abstract void	<u>registerFactory</u> (java.lang.String protocolID, java.lang.String factoryClassName) Registers a factory class for the given protocol.
abstract void	<u>registerImplementation</u> (java.lang.String ID, java.lang.String parserClassName, java.lang.String factoryClassName) Registers a parser and factory implementation using the given ID.
abstract void	<u>registerParser</u> (java.lang.String protocolID, java.lang.String parserClassName) Registers a parser implementation for the given protocol.

TAB. 12

5000

```

void release()
    Shuts down the given SIP UA.
void bindReq(SipEndUser user, SipUserAgentClientListener ssmIc, SipUserAgentServerListener
ssmIs)
    Binds the given user's event listeners to the given SIP UA.
void configureReq(SipManagementListener sml, SipConfigurationReq conf)
    Configures the given SIP UA.
void connectionReq(SipEndUser user, int connectionId, SipConnectionReq invitation,
java.io.Serializable message, java.lang.String mimeType)
    Allows generating a request to establish a session.
void connectionRsp(SipEndUser user, int connectionId, int status, java.io.Serializable message,
java.lang.String mimeType)
    Allows replying to a request to establish a session.
void connectionStatusReq(SipEndUser user, int connectionId, int type, boolean isMiddle,
java.io.Serializable message, java.lang.String mimeType)
    Allows generating provisional responses.
void connectionStatusRsp(SipEndUser user, int connectionId, int type, java.io.Serializable message,
java.lang.String mimeType)
    Allows generating explicitly an ACK.
void disconnectionReq(SipEndUser user, int connectionId, java.io.Serializable message,
java.lang.String mimeType)
    Allows closing a session.
void disconnectionRsp(SipEndUser user, int connectionId)
    Allows replying to an incoming request to close a session.
void optionsReq(SipEndUser user, int connectionId, java.lang.String target, java.io.Serializable body,
java.lang.String mimeType)
    Allows sending OPTIONS.
void optionsRsp(SipEndUser user, int connectionId, java.io.Serializable body, java.lang.String
mimeType)
    Allows replying to an incoming OPTIONS.
void messageReq(SipEndUser user, int connectionId, java.lang.String target, java.io.Serializable body,
java.lang.String mimeType)
    Allows sending a MESSAGE.
void messageRsp(SipEndUser user, int connectionId, java.io.Serializable body, java.lang.String
mimeType)
    Allows replying to an incoming MESSAGE.
void provisionalAckReq(SipEndUser user, int connectionId, java.io.Serializable message,
java.lang.String mimeType)
    Allows sending a PRACK.
void provisionalAckRsp(SipEndUser user, int connectionId, java.io.Serializable message,
java.lang.String mimeType)
    Allows replying to an incoming PRACK.
void registerReq(SipEndUser user, SipRegistrationReq registration, java.io.Serializable body,
java.lang.String bodyType)
    Allows registering the specified user at the given SIP Registrar.
void reservationCoordReq(SipEndUser user, int connectionId, java.io.Serializable message,
java.lang.String mimeType)
    Allows sending an UPDATE.
void reservationCoordRsp(SipEndUser user, int connectionId, java.io.Serializable message,
java.lang.String mimeType)
    Allows replying to an incoming UPDATE.

```

TAB. 13

5100


```
void connectionCfm(SipEndUser user, int connectionId, java.lang.String from, java.io.Serializable  
    body, java.lang.String mimeType)  
    Acknowledges a SIP session establishment.  
void provisionalAckCfm(SipEndUser user, int connectionId, java.lang.String from,  
    java.io.Serializable body, java.lang.String mimeType)  
    Acknowledges a PRACK.  
void registerCfm(SipEndUser user, SipRegistrationCfm registration, int status,  
    java.io.Serializable body, java.lang.String mimeType)  
    Acknowledges a registration request.
```

TAB. 14

5200


```
void connectionInd(SipEndUser user, int connectionId, java.lang.String from,  
    java.io.Serializable body, java.lang.String mimeType)  
    Indicates an incoming request of SIP session establishment.  
void provisionalAckInd(SipEndUser user, int connectionId, java.lang.String from,  
    java.io.Serializable body, java.lang.String mimeType)  
    Indicates an incoming PRACK.
```

TAB. 15

5300


```
void connectionStatusInd(SipEndUser user, int connectionId, int type, java.lang.String typeString,  
    java.lang.String from, java.io.Serializable body, java.lang.String mimeType)  
    Indicates an incoming provisional response from the peer.  
void disconnectionCfm(SipEndUser user, int connectionId, java.lang.String from,  
    java.io.Serializable body, java.lang.String mimeType)  
    Acknowledges a former request to close the session.  
void disconnectionInd(SipEndUser user, int connectionId, java.lang.String from,  
    java.io.Serializable body, java.lang.String mimeType)  
    Indicates a request from the peer to close the session.  
void optionsInd(SipEndUser user, int connectionId, java.lang.String from, java.io.Serializable body,  
    java.lang.String mimeType)  
    Notifies the incoming of OPTIONS.  
void messageInd(SipEndUser user, int connectionId, java.lang.String from,  
    java.io.Serializable body, java.lang.String mimeType)  
    Notifies the incoming of a MESSAGE.  
void optionsCfm(SipEndUser user, int connectionId, java.lang.String from, java.io.Serializable body,  
    java.lang.String mimeType)  
    Acknowledges a former request to send OPTIONS.  
void messageCfm(SipEndUser user, int connectionId, java.lang.String from,  
    java.io.Serializable body, java.lang.String mimeType)  
    Acknowledges a former request to send a MESSAGE.  
void reservationCoordCfm(SipEndUser user, int connectionId, java.lang.String from,  
    java.io.Serializable body, java.lang.String mimeType)  
    Acknowledges a former request to send an UPDATE.  
void reservationCoordInd(SipEndUser user, int connectionId, java.lang.String from,  
    java.io.Serializable body, java.lang.String mimeType)  
    Indicates an incoming UPDATE sent by the peer.
```

TAB. 16

5400


```
void bindReq(SipEndUser user, SipRegistrarListener ssmI)
    Binds the given user's event listener to the given SIP UA.
void configureReq(SipManagementListener sml, SipConfigurationReq conf)
    Configures the given SIP UA.
void optionsRsp(SipEndUser user, int connectionId, java.io.Serializable body,
    java.lang.String mimeType)
    Acknowledges OPTIONS.
void messageRsp(SipEndUser user, int connectionId, java.io.Serializable body,
    java.lang.String mimeType)
    Acknowledges a MESSAGE.
void registerRsp(SipEndUser user, int connectionId, SipRegistrationRsp registration, int status,
    java.io.Serializable body, java.lang.String bodytype)
    Primitive allowing a Registrar to reply to a request for registration from SIP users.
```

TAB. 17

5500


```
void registerInd(SipEndUser user, int connectionId, SipRegistrationInd registration,
    java.io.Serializable body, java.lang.String mimeType)
    Indicates a request from the peer to make a registration.
void optionsInd(SipEndUser user, int connectionId, java.lang.String from, java.io.Serializable body,
    java.lang.String mimeType)
    Notifies the incoming of OPTIONS.
void messageInd(SipEndUser user, int connectionId, java.lang.String from, java.io.Serializable body,
    java.lang.String mimeType)
    Notifies the incoming of a MESSAGE.
```

TAB. 18
(Part 1)

5600

Source State	Triggering event	Guard condition	Action	Target state
Root.Idle	disconnectionCfm		releaseCfm	Root.Idle
"	releaseRsp		disconnectionRsp	Root.Idle
Root.WaitRenewLeaseRsp	Timer T2 expires		failureInd	Root.WaitToClear
Root.WaitRenewLeaseCfm	connectionStatusInd	Status > = 300	abortInd	Root.Idle
"	Timer T101 expires		failureInd	Root.WaitToClear
Root.WaitPreNegotRsp	Timer T1 expires		failureInd	Root.WaitToClear
Root.WaitPreNegotCfm	connectionStatusInd	Status > = 300	abortInd	Root.Idle
"	Timer T102 expires		failureInd	Root.WaitToClear
Root.NegOfferer	Timer T103 expires		failureInd	Root.WaitToClear
Root.ReNegOfferer.WaitNegReq	connectionStatusInd	Status > = 300	abortInd	Root.SessionEstablished
"	Timer T5 expires		ResvMtx.sign _al abortInd	Root.Idle
Root.NegOfferer.WaitTryingInd	connectionStatusInd	Status > = 300	abortInd	Root.Idle
Root.NegOfferer.WaitProgressInd	connectionStatusInd	Status > = 300	abortInd	Root.Idle
Root.NegOfferer.WaitNegStatusRsp	Timer T6 expires		failureInd	Root.WaitToClear
Root.NegOfferer.WaitPrackCfm	connectionStatusInd	Status > = 300	failureInd	Root.WaitToClear
Root.NegOfferer.WaitReservCompl	Timer T7 expires		failureInd	Root.WaitToClear
Root.NegOfferer.WaitRCoordCfm	connectionStatusInd	Status > = 300	failureInd	Root.WaitToClear
Root.NegOfferer.WaitRinging	connectionStatusInd	Status > = 300	failureInd	Root.WaitToClear

TAB. 18
(Part 2)

5600

Source State	Triggering event	Guard condition	Action	Target state
Root.NegOfferer.WaitFinalProvAckCfm	connectionStatusInd	Status > = 300	failureInd	Root.WaitToClear
Root.NegOfferer.WaitConnectionCfm	connectionStatusInd	Status > = 300	failureInd	Root.WaitToClear
Root.NegOfferer.WaitContractAdjusted	Timer T8 expires		failureInd	Root.WaitToClear
Root.ReNegOfferer	Timer T104 expires		failureInd	Root.WaitToClear
Root.ReNegOfferer.WaitReNegReq	connectionStatusInd	Status > = 300	abortInd	Root.SessionEstablished
"	Timer T9 expires		ResvMtx.signal abortInd	Root.Idle
Root.ReNegOfferer.WaitTryingInd	connectionStatusInd	Status > = 300	abortInd	Root.SessionEstablished
Root.ReNegOfferer.WaitProgressInd	connectionStatusInd	Status > = 300	abortInd	Root.SessionEstablished
Root.ReNegOfferer.WaitNegStatusRsp	Timer T10 expires		failureInd	Root.WaitToClear
Root.ReNegOfferer.WaitPrackCfm	connectionStatusInd	Status > = 300	abortInd	Root.SessionEstablished
Root.ReNegOfferer.WaitReservCompl	Timer T11 expires		failureInd	Root.WaitToClear
Root.ReNegOfferer.WaitCoordCfm	connectionStatusInd	Status > = 300	abortInd	Root.SessionEstablished
Root.ReNegOfferer.WaitRinging	connectionStatusInd	Status > = 300	failureInd	Root.WaitToClear
Root.ReNegOfferer.WaitFinalProvAckCfm	connectionStatusInd	Status > = 300	abortInd	Root.SessionEstablished
Root.ReNegOfferer.WaitFConnectionCfm	connectionStatusInd	Status > = 300	abortInd	Root.SessionEstablished
Root.ReNegOfferer.WaitContractAdjusted	Timer T12 expires		failureInd	Root.WaitToClear
Root.NegAnswerer	negotiationReq		abortInd	Root.NegAnswerer
"	reNegotiationReq		abortInd	Root.NegAnswerer

TAB. 18 (Part 3)

5600

Source State	Triggering event	Guard condition	Action	Target state
"	Timer T105 expires		failureInd	Root.WaitToClear
Root.NegAnswerer.WaitAnswer	Timer T13 expires		failureInd	Root.WaitToClear
Root.NegAnswerer.WaitCounterOffer				
Root.NegAnswerer.WaitNegStatusRsp	Timer T14 expires		failureInd	Root.WaitToClear
Root.NegAnswerer.WaitFinalProvAck				
Root.NegAnswerer.WaitFinalAnswer	Timer T15 expires		failureInd	Root.WaitToClear
Root.NegAnswerer.HoldOnAnswer				
Root.NegAnswerer.WaitConnectionCfm				
Root.NegAnswerer.WaitForCoordDone. WaitLocalCoord	Timer T19 expires		failureInd	Root.WaitToClear
Root.NegAnswerer.WaitForCoordDone. WaitRemoteCoord	Timer T20 expires		failureInd	Root.WaitToClear
Root.NegAnswerer.WaitForCoordDone. LocalTokenAchieved				
Root.NegAnswerer.WaitForCoordDone. RemoteTokenAchieved				
Root.ReNegAnswerer	Timer T106 expires		failureInd	Root.WaitToClear
Root.ReNegAnswerer.WaitAnswer	Timer T16 expires		failureInd	Root.WaitToClear
Root.ReNegAnswerer.WaitCounterOffer				
Root.ReNegAnswerer.WaitNegStatusRsp	Timer T17 expires		failureInd	Root.WaitToClear

TAB. 18
(Part 4)

5600

Source State	Triggering event	Guard condition	Action	Target state
Root.ReNegAnswerer.WaitFinalProvAck				
Root.ReNegAnswerer.WaitFinalAnswer	Timer T18 expires		failureInd	Root.WaitToClear
Root.ReNegAnswerer.HoldOnAnswer				
Root.ReNegAnswerer.WaitConnectionCfm				
Root.ReNegAnswerer.WaitForCoordDone. WaitLocalCoord	Timer T21 expires		failureInd	Root.WaitToClear
Root.ReNegAnswerer.WaitForCoordDone. WaitRemoteCoord	Timer T22 expires		failureInd	Root.WaitToClear
Root.ReNegAnswerer.WaitForCoordDone. LocalTokenAchieved				
Root.ReNegAnswerer.WaitForCoordDone. RemoteTokenAchieved				
Root.SessionEstablished				
Root.WaitReleaseRsp	disconnectionCfm		releaseCfm	Root.WaitReleaseRsp
"	Timer T3 expires		releaseReq	Root.Wait DisconnectionCfm
Root.WaitToClear	Timer T4 expires		releaseReq	Root.Wait DisconnectionCfm
Root.WaitDisconnectionCfm	releaseRsp		disconnectionRsp	Root.Wait DisconnectionCfm

57/58

TAB. 19

(Part 1)

5700

Name	Description	Default Value
T1	Waiting pre-negotiation response from E2ENP UA API Service User	TBD
T2	Waiting lease renew response from E2ENP UA API Service User	TBD
T3	Waiting release response from E2ENP UA API Service User	TBD
T4	Waiting release request from E2ENP UA API Service User, after the E2ENP UA has generated a failure indication or a congestion indication to the Service User	TBD
T5	Waiting negotiation request from E2ENP UA API Service User after having booked the resource reservation mutex at the offerer side	TBD
T6	Waiting negotiation status response with counter-offer, from E2ENP UA API Service User at the offerer side	TBD
T7	Waiting start reservation response from E2ENP UA API Service User at the offerer side	TBD
T8	Waiting negotiation status response from E2ENP UA API Service User to handle contract adjustment at the offerer side	TBD
T9	Waiting re-negotiation request from E2ENP UA API Service User after having booked the resource reservation mutex at the offerer side	TBD
T10	Waiting re-negotiation status response with counteroffer, from E2ENP UA API Service User at the offerer side	TBD
T11	Waiting start reservation response from E2ENP UA API Service User, during re-negotiation at the offerer side	TBD
T12	Waiting re-negotiation status response from E2ENP UA API Service User to handle contract adjustment at the offerer side	TBD
T13	Waiting negotiation status response with answer from E2ENP UA API Service User at the answerer side	TBD
T14	Waiting negotiation status response with counteroffer from E2ENP UA API Service User at the answerer side	TBD
T15	Waiting negotiation response from E2ENP UA API Service User at the answerer side	TBD
T16	Waiting re-negotiation status response with answer from E2ENP UA API Service User at the answerer side	TBD

TAB. 19

(Part 2)

5700

Name	Description	Default Value
T17	Waiting re-negotiation status response with counteroffer from E2ENP UA API Service User at the answerer side	TBD
T18	Waiting re-negotiation response from E2ENP UA API Service User at the answerer side	TBD
T19	Waiting start reservation response from E2ENP UA API Service User at the offerer side during negotiations	TBD
T20	Waiting negotiation response from E2ENP UA API Service User side for handling adjusted contracts during negotiations	TBD
T21	Waiting start reservation response from E2ENP UA API Service User at the offerer side during re-negotiations	TBD
T22	Waiting negotiation response from E2ENP UA API Service User side for handling adjusted contracts during re-negotiations	TBD
T101	Waiting OPTIONS confirmation from SIP API for handling lease renewal primitive	TBD
T102	Waiting MESSAGE confirmation from SIP UA for handling pre-negotiation primitive	TBD
T103	Exceeding delay in receiving any SIP UA Generic API primitive while acting as offerer in a negotiation process	TBD
T104	Exceeding delay in receiving any SIP UA Generic API primitive while acting as answerer in a negotiation process	TBD
T105	Exceeding delay in receiving any SIP UA Generic API primitive while acting as offerer in a re-negotiation process	TBD
T106	Exceeding delay in receiving any SIP UA Generic API primitive while acting as answerer in a re-negotiation process	TBD